PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

MAESTRÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

MENCIÓN EN REDES DE COMUNICACIONES

TESIS

TEMA:

"Desarrollo de un prototipo de monitoreo y control de los sistemas de transmisión de Radio o Televisión ubicados en lugares de difícil acceso usando IoT".

AUTOR: Pedro Antonio Collaguazo Reinoso, Ing.

DIRECTOR: Juan Francisco Chafla Altamirano, Ing., MSc.

Quito, Diciembre 2020

DECLARACIÓN

Yo, PEDRO ANTONIO COLLAGUAZO REINOSO, declaro bajo juramento que el proyecto presentado a continuación es de autoría propia; y que el contenido del documento no ha sido presentado por ningún grado o calificación profesional. El contenido de la parte teórica y los temas consultados se encuentran referenciado dentro de este documento.

Pedro Antonio Collaguazo Reinoso

C.I. 171993881-1

DEDICATORIA

Este logro lo dedico a Dios con todo mi corazón, su fortaleza me ayudado a seguir adelante y obtener un logro más en mi vida, su bendición me cuida y protege para seguir avanzando un en mi vida profesional.

También dedico este logro a mi madre María, quien ha sido mi soporte e inspiración, para seguir avanzando en cada momento.

A mi padre y hermanos que siempre me han apoyado en mi carrera, para ser una mejor persona y profesional.

A mi mujer Alexandra y mis hijos Christian, Matteo y Valentina. Que son la mayor bendición que me dio Dios en mi vida, los AMO y siempre serán mi mayor tesoro.

Gracias a toda mi familia

AGRADECIMIENTO

Agradezco a Dios por todo lo recibido en mi vida, por los buenos y malos momentos que he pasado junto a mis seres queridos, y por tenerme de pie aun cuando he caído en varias ocasiones, Dios gracias por estar siempre ahí.

Agradezco a mi madre por todo el apoyo que me han brindado, por siempre motivarme para seguir adelante y mostrarme que, aunque me encuentre derrotado, siempre existe una luz de esperanza para alcanzar una meta.

Agradezco a mi familia por apoyarme para seguir mis metas y objetivos, por ayudarme a cumplirlos y por siempre estar ahí cuando los necesito y ser mi motor para seguir adelante.

Agradezco a mi director de tesis Juan Francisco, por si tiempo, paciencia y apoyo durante toda la etapa de estudios de la Maestría.

Gracias infinitas a todos

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El trabajo presentado en este documento se basa en el diseño y construcción de un prototipo de tecnología que permita monitorear y controlar ciertos parámetros principales dentro de una estación de radio o televisión, haciendo uso de la teoría y aplicación del Internet de las cosas (IoT).

La aplicación con la tecnología del Internet de las cosas (IoT), permitirá la interconexión de dispositivos hacia el Internet, para conocer el comportamiento de una estación repetidora y los equipos que se encuentran operando dentro de este tipo de sistemas y que son importantes en el campo de las telecomunicaciones. La transmisión de señal al aire que este tipo de equipos emiten incluyen programación como noticias nacionales e internacionales, deportes, entretenimiento, cocina y más; hacia poblaciones vulnerables y no vulnerables dentro del territorio ecuatoriano.

La principal característica de estos sistemas se basan en la transmisión de señales de alta potencia dentro del rango de frecuencia que se transmiten los servicios de Radio y Televisión, las frecuencias utilizadas para transmisiones de estos servicios se dividen en cinco sub-bandas (Banda I = 47 – 68 MHz, Banda II [FM] = 87 – 110 MHz, Banda III = 174 - 230 MHz, Banda IV = 470 – 606 MHz, Banda V = 606 – 862 MHz), a causa de la alta potencia que estos sistemas transmiten, deben ubicarse en sitios altos y lejanos a los pueblos o ciudades a las cuales van a brindar cobertura, y así evitar que las señales de radio frecuencia emitidas desde la estación puedan afectar la salud de los habitantes en una población dentro del área de cobertura.

A causa de la problemática mencionada anteriormente, estos sistemas se ubican en sitios donde a veces su acceso es verdaderamente complejo, ya sea por el camino para llegar o por lo lejano que puede encontrarse la ubicación de la infraestructura de uno de estos sistemas. En ocasiones dependiendo de la empresa que brinda los servicios de mantenimiento, se debe viajar días antes para llegar al sitio y a veces se realiza horas de caminata para realizar un reinicio de los equipos en la estación. Bajo esta circunstancia se debe tomar en cuenta que la falla en estos sistemas puede darse desde un simple congelamiento del software del equipo que impide el funcionamiento del equipo y que se soluciona con un reinicio, hasta el daño físico de alguna de las etapas de potencia de un

transmisor y que para dar solución se debe realizar el cambio completo del equipo o la reparación en sitio si es posible. Para cualquiera de estos ejemplos de solución que puede darse es necesario conocer el estado de lo que sucede en la estación y el problema que se produjo para poder dar una respuesta inmediata.

Hasta el momento no existe un sistema que permita conocer el estado de operación de este tipo de equipos de transmisión que se encuentran ubicados en las estaciones remotas y que permita optimizar los tiempos de respuesta a fallas ante cualquier eventualidad. Es por este motivo que se pretende aplicar un concepto de tecnología como es el Internet de las cosas (IoT), para crear un prototipo que permita conocer el estado actual de los equipos en la estación e inmediatamente dar una respuesta rápida, para optimizar los tiempos de respuesta e inmediatamente poder habilitar el sistema y reestablecer la señal al aire.

JUSTIFICACIÓN

La necesidad de la implementación de un prototipo para monitoreo y control en una estación remota de difícil acceso, parte de los problemas que a diario se tiene en las empresas que prestan servicios de mantenimiento a las estaciones repetidoras de radio y televisión; ya que cuando existe un incidente con los equipos de transmisión en la estación; los tiempos de respuesta se prolongan por diversos inconvenientes que se tiene como: cuando el camino de acceso a la estación se encuentra en malas condiciones, el sitio donde se encuentran los equipos requiere caminar largas distancias, el técnico más cercano a la estación se encuentra en otra ciudad o los equipos presenta daños físicos que no pueden ser reparados en el sitio.

Por este motivo se requiere un sistema de monitoreo en tiempo real para conocer el estado de los parámetros principales de operación en una estación repetidora; que puede ser de transmisión de Radio o Televisión.

El prototipo debe permitir mejorar los tiempos de respuesta a fallas, cuando existe alguna eventualidad en la estación.

Con los datos reunidos y el monitoreo constante se puede evitar que una estación quede fuera del aire el menor tiempo posible y poder determinar cuándo se requiere una atención del tipo preventiva y corregir algún tipo de problema a futuro.

Este sistema ayuda a complementar en la red de Internet, equipos de transmisión de radio o televisión que son parte del campo de las telecomunicaciones, y mantener controlados parámetros de importancia en tiempo real dentro de una estación remota. Esta implementación se considera de gran importancia para las empresas que prestan servicio de mantenimiento preventivo y correctivo, permitiendo anticipar el problema en la estación antes de dar la respectiva atención.

Una ventaja que se puede mencionar es la optimización del tiempo respuesta a fallas, optimización de recursos económicos y materiales que pueden ser usados en otras áreas para l desarrollo y mejora de la empresa, y la optimización de personal técnico que pueden trabajar en otros proyectos que se encuentren en ejecución.

ANTECEDENTES

Los sistemas de telecomunicaciones son de vital importancia para la humanidad debido a la comunicación de mensajes que se puede dar a través de este tipo de sistemas. Un área que es importante es la transmisión de señales por un medio de transmisión inalámbrico. Dentro de este tipo de medio se puede asociar la transmisión de señales de Radio y Televisión, que por su naturaleza y la potencia a la que operan estos sistemas sus estaciones se encuentran alejadas de la población.

Para realizar un mantenimiento de las estaciones existentes a nivel nacional, el cumplimiento de este depende de varios factores que intervienen para su correcta ejecución; uno de los factores a mencionar es el tiempo que toma en dirigirse a la estación, realizar el correspondiente mantenimiento y los problemas que tuvo que solucionarse. Para llegar a la estación se debe tomar en cuenta la distancia que se tuvo que trasladar desde oficinas a la estación, puede durar horas de viaje en caso de dirigirse a otra ciudad o minutos en caso de que la atención sea local dentro de la ciudad. También se debe tomar en cuenta si el vehicula llega hasta la puerta de la estación o solo llega hasta una parte del camino de acceso en el cerro y luego se debe realizar caminata para llegar hasta la estación. El tiempo que dure en poner al aire nuevamente la los servicios de radio o televisión en una estación depende de las causas que pudo darse como: estado del clima, daños en la red eléctrica que llega a la estación, picos de voltaje producido por cortes de energía, etc. Cualquier daño en los equipos ocasionado por las causas anteriores, se debe determinar inmediatamente para tomar una decisión inmediata como reparar los equipos afectados en el sitio o proceder a reemplazar el equipo dañado por otro de respaldo, y en el peor escenario dejar la estación fuera de servicio por falta de equipos respaldo.

En la mayoría de las circunstancias se deja la estación fuera del aire debido a la falta de equipos de respaldo, cuando se trata de ciudades donde el tiempo para estar fuera del aire es manejable; por otro lado, en ciudades que tiene mayor índice de población como Quito, Guayaquil, Manabí, Cuenca y otras; se tiene un tiempo de respuesta a fallas de 48H hasta poder poner al aire el servicio. En estas ciudades es importante el cumplimiento del servicio al aire y en el caso de no cumplir con esta disposición el ente regulador envía notificaciones de incumplimiento, llegando hasta el punto de quitar la frecuencia de transmisión por el no cumplimiento de operación del servicio.

En todo este proceso de ejecución de mantenimientos en las estaciones, es primordial el tiempo de respuesta a fallas, ya que se gasta recursos tanto económicos como del área técnica de la empresa encargada o que se encuentra como proveedora de estos servicios.

En un contrato de nivel de servicio (SLA) se establece políticas que satisfaga la calidad de servicio que se ofrece al cliente mediante niveles que deben ser cumplidos estrictamente, dentro del contrato se encuentra los términos y condiciones que deben cumplirse, como por ejemplo el tiempo de respuesta a fallas para cada estación ubicada en las diferentes ciudades, las más importantes un tiempo mínimo, para las de importancia media otro tiempo mayor y para las ciudades no tan importantes un tiempo mayor de respuesta. En el contrato también se puede encontrar las penalidades que se castigan por el incumplimiento de este.

Actualmente, ciertas estaciones ubicadas en las ciudades grandes cuentan con acceso remoto a los equipos para conocer el estado de su operatividad, para el caso de las estaciones ubicadas dentro de las ciudades pequeñas no cuentan con un sistema que permita conocer el estado de los equipos; cuando existe algún problema que deje a la ciudad sin la señal de servicio al aire, la única forma de conocer cuál fue la falla o daño en el sistema es únicamente llegando a la estación.

Otra desventaja que se tiene en los sistemas que no están ubicados en las estaciones de las ciudades principales, es que los equipos son puramente analógicos y antiguos, lo que limita su interconexión hacia un sistema de monitoreo basado en algún tipo de tecnología que permita cambiar la forma de realizar este servicio y optimizar recursos dentro de una empresa.

OBJETIVOS

Objetivo General

Implementar un prototipo para monitorear y controlar los parámetros principales de operación en los equipos dentro de una estación repetidora de difícil acceso usando la plataforma de IoT (Internet of things).

Objetivos Específicos

- Realizar una revisión de investigaciones relacionadas con los sistemas de monitoreo existentes que apliquen el Internet de las cosas (IoT) como parte de la plataforma de transmisión y conocer la influencia que tiene en la actualidad esta tecnología.
- Revisar los sitios de posible instalación del prototipo, así como también el hardware y software posible a utilizar en la construcción del prototipo de monitoreo y control.
- Implementar un sistema de sensores que permita la recolección de información de operación de las estaciones repetidoras de radio y TV dentro de su estructura civil.
- Implementar un sistema de comunicaciones para la transmisión de información desde la estación remota hacia el cliente, para adquirir, almacenar, procesar y tomar decisiones.
- Diseñar un sistema IoT que permita el monitoreo y control en una estación repetidora de radio o televisión.
- Desarrollar una interfaz gráfica que permita la interacción con la estación remota y el usuario para el monitoreo y control de los equipos de la estación repetidora.

CAPÍTULO I

1. INVESTIGACIONES RELACIONADAS

1.1. Análisis de Trabajos Previos

Una vez hecho la revisión de la literatura de los posibles temas afines al prototipo planteado, se puede determinar que los sistemas de monitoreo son esenciales en un sistema de comunicaciones, bien sea para mantener un monitoreo directo de las interfaces y hacer cambios de configuración al momento de resolver un tema de fallas. Existen una cantidad de campos en los que se aplican el monitoreo y control de equipos mediante sistemas remotos sin tener que estar físicamente frente de un equipo.

En la actualidad el uso de dispositivos u objetos interconectados a Internet aplicando la tecnología del Internet de las cosas, los conceptos usados e esta tecnología son ampliamente usado en la ingeniería en proyectos como un monitoreo ambiental, ciudades inteligentes, entre otras (Rodriguez Sotelo, Lopéz Londoño, Vega Botero, & Flóres Hurtado, 2017). El Internet de las cosas IoT contiene una amplia literatura, que permite la interpretación desde varios puntos de vista para la aplicación, el protocolos más conocidos y usados en IoT es MQTT (Message Queue Telemetry Transport), este protocolo se usa para la comunicación Maquina a Maquina que se orienta a la comunicación de sensores, este protocolo ocupa bajo ancho de banda y optimiza el consumo de energía en los dispositivos.

La aplicación de la tecnología con el IoT, se puede usar para sistemas de monitoreo de un sistema de generación fotovoltaica (Flores Cortéz & Rosa, 2016), en prototipos que permitan el monitoreo y control de seguridad para el hogar (Arequipa Cunalata, 2019), monitoreo de cultivos dentro de la parte agrícola (Gómez, Castaño, Mercado, García, & Fernández, 2017), monitoreo de animales dentro del campo agrícola, se puede usar para la recolección de información de parámetros del medio ambiente y poder pronosticar resultados en base al procesamiento de esta información (Quiñonez, González, Torres, & Jumbo, 2017), etc.

Básicamente, usa un modelo de publicación y suscripción; de un lado se tiene varios clientes que se conectan a un bróker central, donde se configura el *topic*[1] al que pertenece cada uno de los sensores agrupándolos de acuerdo a un diseño planteado, un grupo de clientes puede estar formado por dispositivos sensores que envían datos que son procesados y convertidos en información útil para luego ser enviada hacia los clientes que previamente se subscribieron en el bróker y solicitaron un tipo de información recolectada.

De los sistemas de monitoreo que se analizaron en la bibliografía (Arequipa Cunalata, 2019) (Flores Cortéz & Rosa, 2016) (Gómez, Castaño, Mercado, García, & Fernández, 2017) (Quiñonez, González, Torres, & Jumbo, 2017) (Rodriguez Sotelo, Lopéz Londoño, Vega Botero, & Flóres Hurtado, 2017) (Yauri Rodriguez, 2016) (Peña Merizalde & Suquillo Chuquimarca, 2016), ninguno tiene una aplicación en sistemas de radio y televisión ubicados en un cerro para el monitoreo de equipos. Uno de los inconvenientes para implementar un sistema de monitoreo de esta naturaleza es la alta emisión de potencia de transmisión que se tiene cerca de los equipos y por ende la alta contaminación de señales de radio frecuencia existentes en un cerro, generadas por la alta densidad de antenas de transmisión instaladas independientemente del servicio que provee. Los sistemas de IoT revisados en la documentación, no tienen este inconveniente ya que son aplicados en ambientes libres de Radio Frecuencia.

Rodriguez Sotelo, Lopéz Londoño, Vega Botero, & Flóres Hurtado (2017) menciona:

> Un sistema de monitoreo y control remoto de un regulador de presión a través de una herramienta de comunicación IOT, utilizando como plataformas de programación MatLab® y HTML. El sistema de desarrollo para la adquisición y la comunicación corresponde a un Arduino Mega y el Shield de Ethernet respectivamente. Este sistema permite controlar la referencia del regulador de presión y monitorear la información de los transductores de la planta. Entre las ventajas de este tipo de desarrollos es que el servidor es gratuito y el sistema de desarrollo es de bajo costo, además con este tipo de proyectos se puede fortalecer la infraestructura de equipos de laboratorio en el área de ingeniería para poder ser manipulados de forma remota (p. 391).

[1] Topic: es el tema al cual se suscribe el cliente para el tipo de información que requiere, este tema contiene la información de los datos de un sensor.

Quiñonez, González, Torres & Jumbo (2017) proponen:

> Un sistema para la recolección de datos meteorológicos usando una Red de Sensores Inalámbricos (RSI), capaz de transmitir los datos en tiempo real. El sistema logra automatizar los procesos de obtención de datos de manera continua y a largo plazo, por medio de un módulo de abastecimiento de energía solar que permite autonomía para su funcionamiento. Para la viabilidad del diseño e implementación de prototipos se propone la construcción de dos sistemas basados en DigiMesh y Wi-Fi, los que se pueden aplicar a diferentes escenarios como zonas urbanas y rurales. Adicionalmente se evalúa la transmisión de información a plataformas de Internet de las Cosas (IoT), en donde se gestionará y visualizará los datos obtenidos por los nodos. Este sistema fue concebido como una alternativa de bajo costo comparado con estaciones meteorológicas convencionales que posean estas prestaciones y está basado en componentes de hardware y software libre (p. 329).

Yauri Rodriguez (2016) en su trabajo presenta:

> Un diseño, desarrollo y construcción de un sistema de monitoreo remoto, basado en IOT, para la adquisición, procesamiento, envío y visualización de datos de señales electrocardiográficas (ECG). El sistema implementa un módulo sensor embebido con comunicación inalámbrica que es capaz de transmitir datos por medio de la tecnología WI-FI o GPRS. El control de la adquisición, procesamiento y envío de las lecturas obtenidas desde el nodo sensor se realiza con un microcontrolador PIC24 de bajo consumo sobre el cual se implementa un sistema de control para el ahorro de energía aumentando la autonomía y funcionamiento del módulo. Los datos adquiridos por el microcontrolador se procesan y se adaptan al medio de comunicación sobre el cual se desea transmitir. Los datos se envían a un servidor Web en Internet usando la tecnología GPRS o mediante una red WiFi. En el servidor, se desarrolló una aplicación Web que crea conexiones basadas en Websockets para recibir la información del módulo sensor embebido y luego mostrar las señales adquiridas a todos los usuarios conectados en tiempo real. Este sistema permite que la señal ECG de la persona que es monitoreada sea supervisada de manera continua para prevenir eventos que puedan poner en riesgo su salud (p. 94).

Flores Cortéz & Rosa (2016) en su artículo presentan:

> Los resultados obtenidos de la investigación denominada “Internet de las cosas” interconexión de un sistema de generación fotovoltaico para su monitoreo desde la nube”, desarrollada durante 2015 en la Utec, y cuyo objetivo es el diseño e implementación de un sistema electrónico de bajo costo que permitiera el monitoreo vía Internet del voltaje producido por un panel solar (p. 40).

Arequipa Cunalata (2019) en su tesis presenta:

> Un prototipo de sistema de seguridad que utiliza protocolos del Internet de las Cosas (IoT) sobre la plataforma Zolertia Remote, el mismo que busca conocer el desempeño de estas tecnologías dando una alternativa de solución al problema de inseguridad. El prototipo consta de tres secciones: una red inalámbrica de sensores, una implementación de bróker o servidor privado MQTT y una aplicación móvil Android. La red inalámbrica de sensores consiste en tres motas Zolertia Remote, dos de éstas actúan como nodos sensores y la tercera como un nodo gateway. Los nodos sensores detectan el movimiento mediante sensores PIR y transmiten esa información de forma inalámbrica al nodo gateway. La sección bróker privado consiste en la combinación del nodo gateway con un Raspberry Pi 3, combinación que conforma el enrutador de borde de la red; esta sección también implementa scripts desarrollados en Python, los mismos que permiten la comunicación con un bróker público. La aplicación móvil permite recibir la información recolectada por la red inalámbrica y también interactuar con el sistema remotamente (p. XVI).

Peña Merizalde & Suquillo Chuquimarca (2016) en su tesis presentan:

> El estudio del modelo de referencia del Internet de las Cosas y la implementación de un prototipo domótico que cumpla con los requisitos básicos del modelo; el prototipo se ha realizado como si se tratase de una aplicación de gestión real para una “casa inteligente”, que consta de tres sistemas importantes: seguridad, iluminación y climatización; con sus respectivos sensores (temperatura, luz, etc.) y actuadores (cortinas, ventilador, zumbador, etc.) (p.XX).

Gómez, Castaño, Mercado, García, & Fernández, (2017) presentan en su trabajo

> Un sistema de Internet de las cosas para el monitoreo de cultivos protegidos, a través del desarrollo de un sistema con capacidad de recolectar información de parámetros relacionados con el desarrollo y crecimiento de los cultivos. Los datos

> obtenidos son enviados al servidor para ser procesadas y enviadas al usuario a través de los protocolos y procedimientos del Internet de las cosas (IoT). El propósito es recopilar datos en tiempo real para analizarlos y permitan la toma de decisiones por parte del mismo sistema y el agricultor. El usuario puede interactuar con el sistema de manera remota y recibir las alertas y condiciones especificadas. Los resultados iniciales demuestran que el sistema provee completa información del estado de estos parámetros, ayudando en la tarea del manejo de este tipo de cultivos (p.24).

Las varias soluciones aplicadas en el IoT, para cada una de las aplicaciones presentan varias plataformas sobre las cuales se realiza la programación de los protocolos comunicación pertinentes para la conexión entre los clientes y el bróker central donde se maneja la información recolectada por sistemas sensores, se procesa y se envía.

Una variable importante para estudiar es la cantidad de clientes que se conectan hacia el bróker central para la recolección de información, ya que deberá soportar varios sensores que puedan recolectar gran cantidad de información, procesarla en una base de datos y poder usarla para enviar al subscriptor que requiere cierto tipo de información.

El tipo de software con el que trabaja IoT para la programación es de fácil implementación y para todos los sistemas estudiados que fueron implementados es libre, aunque se puede hablar de ciertos tipos de software que son licenciados, mismos que ofrecen características adicionales que pueden complementar en ciertas funcionalidades.

Los sistemas de monitoreo implementados no siguen un estándar en la implementación, usan diferentes tipos de plataformas que son configurados de diferente manera para la conectividad de los equipos hacia el bróker central, para el caso de los sistemas sensores varios son implementados con diferentes tipos de hardware como por ejemplo Arduino. Para la visualización de los parámetros medidos por los sensores se puede configurar una página web, una aplicación para visualizar la información en un dispositivo Android.

Los parámetros de medición para cada ambiente de aplicación son importantes, ya que en ciertas mediciones puede transmitirse varia información lo que complicaría el funcionamiento y latencia de la transmisión de datos para poder procesar y convertirla en información útil.

Uno de los conceptos importantes que aportan al proyecto es el Internet de las cosas; ya que bajo este tipo de tecnología se realiza la interconexión de los dispositivos sensores

que se utilizara con las interfaces graficas que muestran la información de los parámetros de operación de los sistemas para un constante monitoreo.

Otro concepto es el tipo de protocolo que se utiliza para la publicación y suscripción de los clientes en el bróker central, el protocolo mencionado es MQTT (Message Queue Telemetry Transport). Adicional se debe tener en cuenta que se requiere la configuración de un servidor MQTT público que permita la interconexión con el bróker central.

1.2. Revisión de artículos referentes en el Reglamento y Ley de Radiodifusión y Televisión

Se puede mencionar unos artículos que corresponden a esta área de Radiodifusión y Televisión, que están contenidos dentro del Reglamento y la Ley de Radiodifusión y Televisión, dichos artículos que intervienen directamente con el proyecto planteado y que pueden afectar directamente al concesionario de la frecuencia de una Radio o Televisión. A continuación, se detallan los artículos descritos:

1.2.1 Ley de Radiodifusión y Televisión

Tomando en cuenta que dentro del país debe existir un ordenamiento legal para el uso del espectro radioeléctrico, y también para los servicios que se brindan dentro de este; como el de Radio y Televisión, con el propósito de fomentar la superación técnica, económica y cultural, para tener un desarrollo nacional y una gran evolución tecnológica universal.

La ley de telecomunicaciones fue creada para garantizar el uso óptimo de este recurso en base a normativas y reglas que se deben cumplir para poder hacer uso de un espacio dentro del espectro radioeléctrico.

Un artículo que se puede mencionar dentro de la Ley de Radiodifusión y Televisión es el siguiente: TITULO VI (Del término de las concesiones), el Artículo 67 dice que la concesión de canal o frecuencia para la instalación y funcionamiento de una estacion de radiodifusión y televisión, termina por reincidencia en faltas de carácter técnico que hubieren sido sancionadas con dos multas y una suspensión, mencionado en el literal (e) de este artículo (Ministerio de Telecomunicaciones, 2009).

Se puede mencionar que este artículo es aplicado una vez que la estacion repetidora haya tenido dos multas y una suspensión a causa de fallas técnicas, por ejemplo, que los equipos

se encuentren dañados y no se esté ofreciendo señal del servicio dentro del área de cobertura correspondiente, y que no hayan podido darse solución dentro de un plazo establecido.

No habrá lugar a la reincidencia si el Instituto Ecuatoriano de Telecomunicaciones otorga al concesionario un plazo que no excederá de seis meses para el arreglo definitivo del problema técnico, sin perjuicio de que se ordene la suspensión del funcionamiento de la estación durante el plazo de prórroga (Ministerio de Telecomunicaciones, 2009).

1.2.2 Reglamento a la Ley de Radiodifusión y Televisión

Del Reglamento a la Ley de Radiodifusión y Televisión, se puede analizar el articulo tomado del CAPITULO XIX que habla DE LAS INFRACCIONES Y SANCIONES, en el Articulo 80, CLASE V, literal (a) menciona que, si se suspende las emisiones de una estación por más de 180 días consecutivos, sin autorización de la Superintendencia de Telecomunicaciones (Ministerio de Telecomunicaciones, 2008). Así mismo de la sanción que se aplica se estipula en el mismo Reglamento, Articulo 81, dice que las sanciones se aplicaran de acuerdo con la clase de acción cometida conforme se indica que para las infracciones de CLASE V, se aplicara la sanción de cancelación de concesión, mediante la terminación de contrato y reversión de la frecuencia al estado (Ministerio de Telecomunicaciones, 2008).

Una vez revisado los artículos de la Ley de Radiodifusión y Televisión, se puede ver que es de interés tomarlos en cuenta en el caso de existir problemas de fallas en la estación, mismos que se pueden prolongar durante largos lapsos de tiempo debido a problemas de daños en los equipos. A causa de las fallas y daños en la estación, al concesionario del servicio se aplican multas y hasta la suspensión del servicio por tiempo determinado, y en caso de reincidencias graves se procederá a terminar el contrato de concesión de frecuencia y esta podría ser retirada de la persona que puede ser natural o jurídica, nacional o internacional.

A causa de los inconvenientes que pueden presentarse se ha propuesto el proyecto planteado del sistema de monitoreo, con el que se puede prevenir daños en los equipos que pueden durar meses dependiendo de los repuestos, ya que algunos pueden estar descontinuados debido a la antigüedad de los equipos de transmisión que operan en las estaciones a la actualidad. En base a los datos obtenidos de los sensores se puede realizar la toma de decisiones para ejecutar la acción correspondiente y programar mantenimientos preventivos para solucionas cualquier anomalía en los sistemas de transmisión.

1.3. El Internet de las Cosas (IoT)

1.3.1 IoT en la actualidad

En la actualidad se habla mucho del Internet de las cosas, y de varias aplicaciones que se pueden implementar en diferentes ambientes, se puede mencionar que el Internet de las cosas es una tecnología con un gran futuro y que ha llegado para quedarse, antes se hablaba de domótica, que es la creación de casas inteligentes que son monitoreadas y controladas bajo una plataforma en particular; ahora la domótica se dice que forma parte de la tecnología del Internet de las cosas.

La tecnología del Internet de las cosas y la implementación de sus dispositivos actualmente permite que podamos conectar el mundo físico con lo digital para poder monitorear y controlar, con el objetivo de optimizar recursos y mejorar la eficiencia dentro de una empresa u organización.

Existen varias definiciones para el Internet de las cosas, una de ellas dice que es la interconexión de varios dispositivos o equipos hacia el Internet como se puede ver en la Figura 1; y que mediante el monitoreo se puede verificar el funcionamiento y el estado de operación, mediante reportes de alarmas que se generan a lo largo del tiempo de funcionamiento.

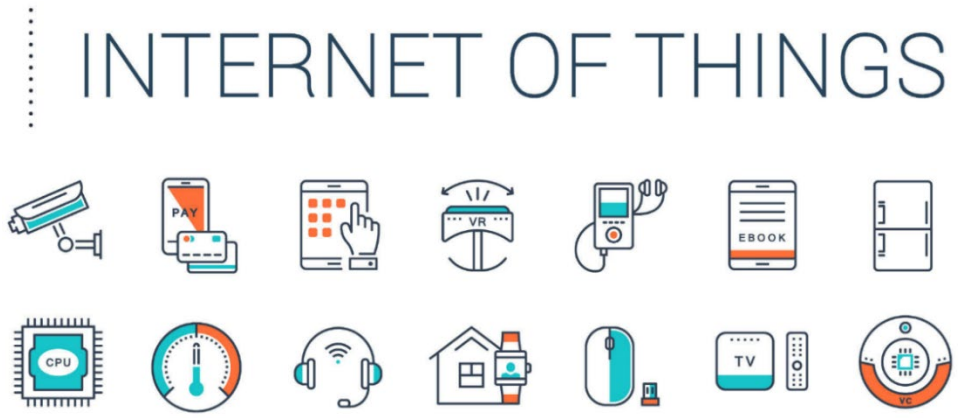

Figura 1. Internet de las Cosas IoT, (Leal, 2019)

La tecnología IoT tiene la capacidad de convertir cualquier equipo u objeto inerte en inteligente dándole una identidad propia en base al uso de sensores que dependiendo de su tipo pueden usarse para una infinidad de aplicaciones, ya que estos tienen la capacidad

de realizar mediciones del estado y los cambios que se pueden producir en su entorno, estos sensores realizan la recolección de información acerca del funcionamiento y generan una base de datos que a la vez son analizados, procesados y usados para una toma de decisiones.

Actualmente, el Internet de las cosas ha cambiado la forma en que las personas o usuarios se relacionan con el mundo físico, haciendo que a través de aplicaciones de IoT sea más fácil el uso de servicios en los diferentes sectores como la agricultura y ganadería, el comercio, hostería (Restaurantes) y otros sectores importantes.

1.3.2 Tecnología 5G como una base de IoT

Una de las principales tecnologías que es importante para el desarrollo de IoT es la tecnología 5G, su evolución se muestra en la Figura 2; desde sus inicios hasta la actualidad en el 2020; ya que puede implementarse en una gran variedad de dispositivos actualmente y conociendo que es una infraestructura en la que se va a transmitir un gran volumen de datos no estructurados y en tiempo real, la tecnología 5G ofrece una alta velocidad de transmisión de información igual 1Gb por usuario de manera simultánea.

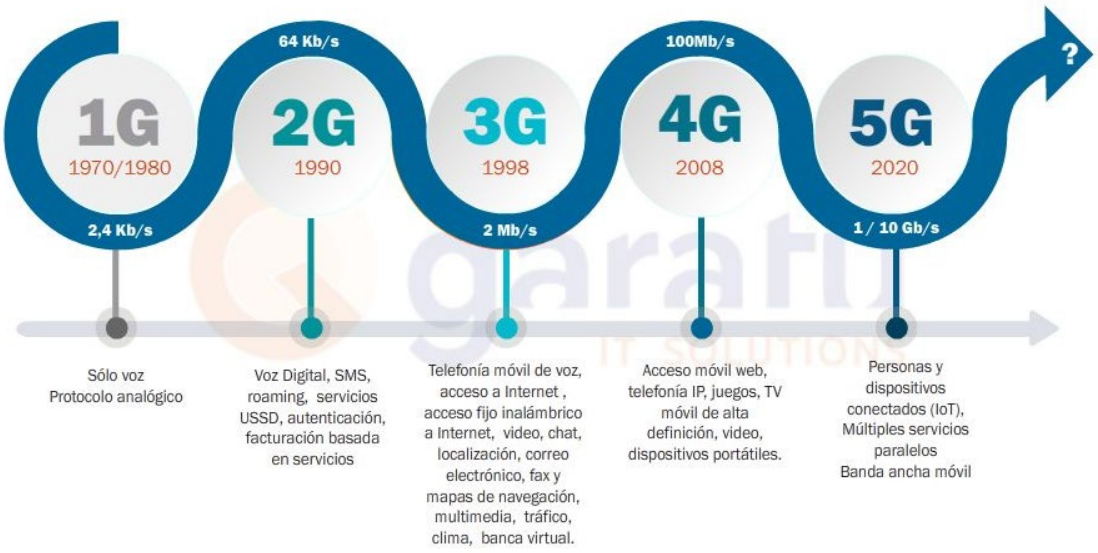

Figura 2. Tecnología 5G base de IoT, (Grupo Garatu, 2019).

La tecnología de quinta generación 5G, se hace relevante por su arquitectura prometedora de adaptarse a casi cualquier espacio de uso dentro del internet de las Cosas, con una flexibilidad altamente diseñada, un futuro cada vez más cercano, se puede podría apreciar

que miles de millones de dispositivos estarán conectados de manera permanente (Grupo Garatu IT Solution, 2019).

1.3.3 Dispositivos para IoT

Los dispositivos utilizados para IoT se pueden encontrar fácilmente en el mercado, desde dispositivos no profesionales hasta dispositivos de alta eficiencia para realizar la programación de toda la plataforma de IoT.

1.3.3.1 *Sistemas para recolección de Información*

Entre los varios tipos sistemas utilizados para recoger los diferentes tipos de información y enviarlos a la nube se pueden describir los siguientes:

1.3.3.1.1. *Sensores Y Actuadores*

Sensores

Se puede comenzar por analizar los sensores para Internet de las cosas que son una parte importante dentro de la habilitación de esta plataforma, los dispositivos sensores son los encargados de recoger todos los datos necesarios que se va a transmitir a través de la plataforma de IoT.

Un sensor se compone de un transductor que cumple la función de medir, detectar e indicar cambios que se originan en un determinado sitio, los cambios se producen en magnitudes físicas como temperatura, humedad, presión y otras que se pueden medir con el uso de un tipo de transductor. Todas las mediciones se transforman en señales eléctricas que van a un acondicionador de señal, luego se amplifica y se convierte en una señal con formato de lectura simple (Envira, 2019).

Un transductor por si solo son inútiles, por este motivo se encuentran formando parte de un sistema electrónico que puede tener una fuente de alimentación, módulo de memoria, controladores, etc.; todos estos elementos se conectan entre sí para recopilar información en forma de datos, luego procesar y enviar dentro de una red para uso de esta información (Envira, 2019).

Las aplicaciones usadas para leer información están determinadas por los tipos de sensores que se utilicen acorde con su funcionalidad, algunos se muestran recogidos por tipos en la imagen siguiente como: las fuentes de alimentación, la conectividad y de forma ocasional, una interfaz de usuario que puede ser una aplicación móvil por ejemplo (Envira, 2019).

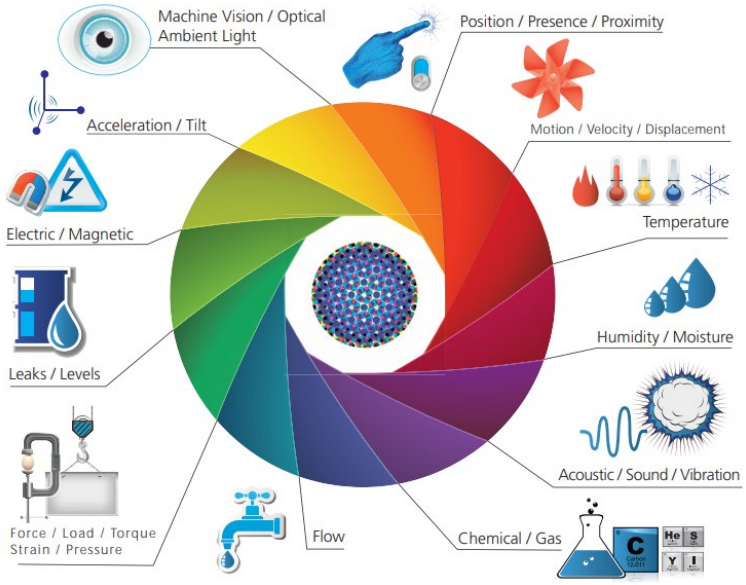

Figura 3. Tipos de sensores para IoT, (Postscapes, 2019).

En la Figura 3 se muestra los diferentes tipos de sensores que se pueden tener para la recolección de información, la utilización de cada uno depende del del ambiente dentro del cual se va a trabajar.

Los sensores deben trabajar con un interfaz que permita convertir la información leída por el sensor en datos que se puedan transmitir hacia la plataforma de IoT y ejecutar todos los procesos necesarios para aprovechar al máximo esta información en el desarrollo y productividad de una empresa.

Actuadores

Se puede definir como dispositivos capaces de transformar la energía eléctrica en una acción como abrir o cerrar un contacto, esto permite automatizar cualquier proceso. Los actuadores más comunes que se pueden encontrar son los relés, indicadores luminosos, motores y electroválvulas.

A los dispositivos actuadores de IoT encargados de detectar captaciones eléctricas se los conoce como Smart Sensors o en español sensores inteligentes, ya que a más de obtener un parámetro físico pueden realizar otras funciones. A más de la inteligencia de estos módulos electrónicos, poseen la funcionalidad de comunicación inalámbrica, lo que les permite interactuar de forma autónoma con otros dispositivos e inclusive cuando se encuentran en movimiento, facilitando su conexión dentro de una red inalámbrica sin necesidad de cablear (Ruiz, 2020).

1.3.3.1.2. *Etiquetas Inteligentes Smart Tags*

Actualmente las empresas requieren conocer los costos del proceso de producción para poder documentarlo, por lo que se debe dar respuestas a preguntas como: ¿Dónde y cuándo se ha producido el producto?, ¿Quién lo ha producido?, ¿Qué materiales han sido utilizados? (Ruiz, 2020).

Para contestar las preguntas anteriores Ruiz (2020) dice que se requiere capturar datos donde no es posible hacerlo humanamente y hacer el seguimiento de tiempos, con su traza y características añadidas. Esta tarea, supone un gran reto sobre todo cuando la velocidad de los procesos es alta y se requiere conocer todos los aspectos relacionados con la fabricación del producto en tiempo real.

Por lo que se requiere de una tecnología sin contacto denominada contactless, misma que permitirá relacionar las señales eléctricas de las máquinas y las mediciones de los sensores con el producto y posteriormente conocer cómo fue cada punto del proceso productivo. Para conocer todo esto se dispone de las opciones siguientes:

Etiquetas impresas

Este tipo de etiquetas se pega o se imprime en el producto para que lo identifique de manera única por categoría o tipo de producto, las más conocidas son los códigos QR o el código de barras (Ruiz, 2020).

Para leer estos códigos se instala lectores ópticos en un punto de interés donde la información leída se envía a un ordenador donde se almacena en una base de datos toda la información del proceso junto a la identificación; estos códigos poseen una limitante que es al momento de realizar la lectura, deben mantener contacto visual directo entre la etiqueta y el lector que no siempre es posible, la otra limitante es que no se puede almacenar información y por lo tanto la única manera de distinguir entre unidades de producto de un mismo tipo es mediante la utilización de contadores de piezas e impresoras adicionales (Ruiz, 2020).

Etiquetas RFID

Las etiquetas RFID son utilizadas cada vez más debido a las limitaciones que tiene los códigos de barras y QR, en esta ocasión se usan tags RFID que son tarjetas de identificación por radiofrecuencia, poseen un microchip con una antena que permite leer información mediante un lector/escritor RFID, posee la ventaja de no requerir contacto

visual, ya que gracias a la memoria del microchip RFID se almacena un número de identificación por cada unidad fabricada y adicional se puede almacenar datos del proceso, de esta manera se podrá conocer el proceso de creación de cada unidad cuando se requiera utilizando un lector (Ruiz, 2020).

Con el abaratamiento de costos de este tipo de etiquetas de forma exponencial, el uso de estas etiquetas es más amplio en diferentes campos, hoy en día son usados en apertura de vehículos, controles de accesos, telepeaje, transporte público, inventario en tiendas, identificación de cajas y pallets, identificación de animales, control de mercancías, sistemas de alarma, localización de pacientes en hospitales, etc., (Ruiz, 2020). Los diferentes tipos de etiquetas RFID disponibles se muestran en la Figura 4.

Los tags RFID se puede decir que difieren dependiendo si son activos que quiere decir que usan baterías para el funcionamiento o pasivos que no usan baterías, el alcance máximo para los tags RFID pasivos es de 10m y para los tags RFID activos pueden superar los 250m. Los tags activos pueden incorporar sensores que pueden ser útiles para cuando los objetos a localizar tienen movimiento (Ruiz, 2020).

Figura 4.- Distintos tipos de TAGS RFID *(Ruiz, 2020)*.

Para el uso de un tipo de etiqueta u otro depende del tipo de material del tag, distancia entre etiquetas y tags, numero de dispositivo, tamaño de los objetos que se requiere etiquetar, etc.; por lo que se requiere realizar pruebas de campo con el objetivo seleccionar los tags más apropiados para el uso en cada situación y así conocer con precisión las limitaciones (Ruiz, 2020).

Luego de seleccionar los lectores y tags, se procede con la conexión de estos a un servidor que se encuentre dentro de la red de una organización quien se beneficiara de la información leída y luego poder almacenar estos datos, el servidor será el encargado de publicar la información obtenida de las etiquetas para que puedan ser almacenados dentro de una plataforma que permita a los clientes suscribirse para conocer toda la información necesaria (Ruiz, 2020).

1.3.3.1.3. *Sistemas de Control Embebidos*

Estos sistemas de control embebidos son aquellos que han sido creados o fabricados para un propósito especifico y realizar funciones dedicadas en tiempo real que cumplen un amplio rango de necesidades (Ruiz, 2020).

Los sistemas de control deben ser programados y luego se deben ajustar para cumplir una tarea específica a la ha sido concebida, por lo que su costo es significativo cuando se requiere adquirir cualquiera de estos componentes (Ruiz, 2020). En la Figura 5, se puede ver las características en capacidad y procesamiento de tarjetas que son usadas para diferentes tipos de proyectos.

Actualmente, existe en el mercado, módulos electrónicos de bajo costo como las tarjetas de Arduino y Raspberry Pi que son usados en el diseño de sistemas embebidos de cualquier tipo para desarrollar proyectos tecnológicos, estos módulos poseen una gran cantidad de accesorios que se conectan directamente como: cámaras, módulos de conexión de red inalámbricos, sensores, memoria, etc., a esto se complementa con librería de software que facilitan la implementación de cualquier solución que se requiera (Ruiz, 2020).

Specs	Arduino Uno 	Raspberry Pi Model B+ 
CPU type	Microcontroller	Microprocessor
Operating System	None	Linux (usually Raspbian)
Speed	16 Mhz	700 Mhz
RAM	2KB	512MB
GPU/Display	None	VideoCore IV GPU
Disk	32KB	Depends on SD card
GPIO pins	14 digital pins (includes 6 analog)	26 digital pins
Other connectivity	None	USB, Ethernet, HDMI, audio
Power consumption	0.25W	3.5W

Figura 5. Arduino vs. Raspberry Pi Modelo B+ *(Ruiz, 2020)*.

La s necesidades que se requiera a nivel de software como velocidad de procesamiento, tamaño de memoria para almacenar datos, cantidad de entradas y salidas, etc., dependerá del tipo de proyectos a implementar; por lo que será conveniente emplear un dispositivo que este acorde con las características principales mostradas en la Tabla 1. Se tener el conocimiento claro de que funciones va a realizar nuestro sistema embebido (Ruiz, 2020).

PROYECTO	ARDUINO	RASPBERRY PI
Servidor Web, FTP, multimedia, etc.		X
Lectura de sensores.	X	
Control de motores, servos, actuadores.	X	
Procesamiento de imágenes.		X
Domótica.	X	X
Dron	X	
Reproductor de tonos	X	
Reproductor de música		X

Tabla 1. Elección del dispositivo de acuerdo a las características (Elaboración propia)

1.3.3.2 *Plataformas de IoT*

Una Plataforma de IoT es el núcleo de esta tecnología, es aquí donde los dispositivos de interconectan e intercambian toda la información recogida. Una plataforma web integrada al Internet de las cosas es el software que conecta el hardware, clientes que se involucran en la obtención de datos como sensores y actuadores, clientes que se suscriben a la plataforma y redes de datos, a este software es al que habitualmente el usuario se conecta para disfrutar del servicio.

Hay que tener en cuenta que no todas las plataformas son precisamente para IoT, pero que generalmente existe cuatro plataformas que se refieren a menudo a las plataformas IoT (Cárdenas, 2016).

1.3.3.2.1 Plataformas de conectividad para IoT

1.3.3.2.1. *Plataformas de conectividad / M2M*

Este tipo de plataforma se basa en la conectividad de dispositivos que se encuentran en una red de telecomunicaciones, rara vez se enfoca en el procesamiento de la información y el conocimiento de los conjuntos de datos de los sensores que se encuentran en línea (Cárdenas, 2016).

1.3.3.2.2. *Backends. IaaS:*

La infraestructura como servicio proporciona alojamiento y potencia de procesamiento para aplicaciones y servicios, estos backends suelen ser optimizados para el uso de aplicaciones de escritorio y móviles, pero IoT ahora también está en foco (Cárdenas, 2016).

1.3.3.2.3. *Plataformas de software específicos de hardware*

Existen empresa que han construido su propio backend y que prácticamente se refieren como una plataforma de IoT, pero existe el inconveniente que la plataforma no está abierta a nadie más en el mercado y por tanto se discute si se debe definir como una plataforma de IoT (Cárdenas, 2016).

1.3.3.2.4. *Extensiones / software para empresas de consumo*

Este tipo se plataformas se basan en software empresariales con sistemas operativos Windows 10, que permiten cada día más la integración con dispositivos de IoT. A menudo estas extensiones no son lo suficientemente avanzadas como para clasificar que es una plataforma IoT, pero puede ser pronto (Cárdenas, 2016).

1.3.4 Aplicaciones importantes

Se puede hablar del Internet de las cosas como una tecnología que se puede aplicar en diferentes campos de la Industria, debido a la gran evolución que ha mantenido durante la última década. Varios dispositivos conectados a la gran Red Internet para facilitar la vida del cliente y mejorar su experiencia mediante el uso de dispositivos que funcionan con esta tecnología.

Los dispositivos IoT funcionan a través de Internet, para controlar funciones de objetos como electrodomésticos, vehículos, accesos, activaciones de circuitos de alarmas.

Actualmente se pueden ver varios dispositivos que funcionan con aplicaciones creadas para diferentes tipos de propósito, entre los que podemos mencionar como ejemplo es el Internet de las cosas aplicados al campo en el área de la agricultura; aquí los dispositivos permiten a los agricultores conocer mediante la recolección de información en forma de datos, el estado de su cultivo en tiempo real para poder realizar una acción en particular si existe inconvenientes con el cultivo. Conocer toda esta información para poder ejecutar una acción correctiva a tiempo puede tener un impacto positivo dentro de la industria, todos los procesos son automatizados lo que conlleva a dedicar menos tiempo en los terrenos y dedicarse más tiempo al estudio de sus cultivos.

La información recolectada se realiza el análisis de datos para optimizar los recursos como el agua de riego y fertilizantes, reduciendo el coste y mejorando la calidad del producto.

Así también se puede mencionar que el inicio del Internet de las Cosas en la vida diaria de los consumidores, se puede apreciar con la creación de dispositivos de uso para los seres humanos como los wearables que son usados para monitorear contantemente signos vitales, y que a través de las funciones de control y de gestión de los sensores situados en casas inteligentes o en los vehículos conectados ha permitido encontrar soluciones a avances en el campo de IoT aplicado en el ámbito industrial.

Los wearables o tecnología vestible son dispositivos electrónicos e inteligentes que se incorporan en la ropa como se aprecia en la Figura 6, con el fin de monitorear varios parámetros de salud y recolectar esta información; entre algunos parámetros se puede tener el ritmo cardiaco, bio-señales musculares y ondas cerebrales; estos datos son de mucha importancia en los seres humanos en especial en un adulto mayor para mantener un registro de sus signos vitales y poder garantizar el bienestar de la persona.

Los wearables son dispositivos que registran lo que vemos como: nuestro ritmo respiratorio, las horas que dormimos e, incluso, nuestra sensación de bienestar, nuestras pulsaciones, o de estrés; refuerzan buenos hábitos de alimentación y salud. Dentro de la industria se ofrece diferentes tipos de wearables: desde muñequeras, pendientes, collares, y anillos hasta prendas de vestir. Y por supuesto, relojes. (Iberdrola, 2018). Estos dispositivos son los más exitosos como se puede ver en la Figura 6.

Figura 6. Wearables (Tecnologia Vestible), (Iberdrola, 2018).

En la figura 7, se puede hablar de otro tipo de tecnología algo parecida a los wearables y que también se encuentra disponible, son los swallowables. Son utilizados la misma área para la salud, pero estos son dispositivos tragables y una vez en el interior se puede recibir, gestionar y obtener información que sirva para una toma de decisiones. Sirven para monitorear enfermedades y mantener el control de pacientes crónicos. También existen píldoras de autentificación que una vez tragadas convierten a al cuerpo humano en un identificador digital.

Como se puede ver acerca de la tecnología wearables, los avances que ha tenido para ser usados como un artículo de vestir. Se puede ver claramente la tendencia que tiene a futuro este tipo de dispositivos que cada vez son más usados por el ser humano, ya que, gracias a la miniaturización de los componentes electrónicos, el desarrollo de los protocolos de comunicación la geolocalización y el software de gestión de datos, su uso tiene mayor demanda, ya que, a diferencia de los teléfonos celulares o Smartphones, estos dispositivos se pueden integrar en los cuerpos de las personas.

Figura 7. Tipos de Wearables en la industria, (Iberdrola, 2018).

CAPÍTULO II

2. REVISIÓN DE SITIOS PARA POSIBLE INSTALACIÓN Y SOLUCIONES DE HARDWARE Y SOFTWARE PARA EL DISEÑO

En este capítulo se realizará el análisis de los sitios remotos donde se podrían realizar la instalación del prototipo de monitoreo y control, para probar el funcionamiento y determinar los problemas que se pueden presentar durante las pruebas.

Para el caso de fotografías de las estaciones, no se especificar el concesionario al que corresponde las fotografías por temas de seguridad y protección de bienes, solo se tomaran como ejemplo para la explicación de los componentes que se pueden encontrar en una estación real, y las condiciones a las que se expone un prototipo dentro de ese ambiente.

Se tomará en cuenta el cableado y las posibles protecciones que se debe tener instalado junto con el prototipo para garantizar la operatividad en caso de existir problemas con la red eléctrica, y que el sistema de monitoreo siga operando sin interrupción.

Para el caso de las posibles soluciones que se puedan implementar para el diseño, se tomara en cuenta conceptos acerca del Internet de las cosas y su funcionamiento mediante las políticas o protocolos que funcionan para esta tecnología. Con el fin de proponer el hardware y software adecuado para poder interconectar estos equipos, y poder tener el control en base al monitoreo.

2.1. Estado de sitios remotos

La implementación de un sistema de monitoreo en un sitio remoto facilita la reparación y habilitación del sitio que se encuentra fuera del aire, por motivos de condiciones climáticas o factores que se ocasionan en el sitio donde se encuentran instalados los sistemas de Radio o Televisión. En ciudades grandes como Quito, Guayaquil y Cuenca, las repetidoras se encuentran instaladas en cerros cercanos y el tiempo estimado en llegar al sitio donde se encuentra instalado el servicio de Radio o Televisión, es de una hora o menos. En

cambio, para las repetidoras que se encuentran instaladas en ciudades pequeñas, el tiempo para trasladarse al sitio aumenta debido a la ubicación.

Una de las topologías comunes para una red de radiodifusión o televisión se puede observar en la figura 8, donde una estación matriz envía su señal de programación a través de un espacio satelital contratado, en un satélite que tenga presencia con su huella satelital dentro del territorio ecuatoriano. Esto facilita llegar con la señal de programación a cualquier sitio ubicado dentro del territorio nacional, una vez que se suba la señal al satélite se puede bajar la señal en cualquier sitio para instalar un repetidor en cualquier ciudad que se requiera.

Este tipo de topologías es utilizado para las señales que transmiten radio y televisión, en especial señal de televisión debido al ancho de banda que se requiere. Las señales de radio por lo general se multiplexan con las de televisión, y se envían en el mismo ancho de banda para optimizar los recursos.

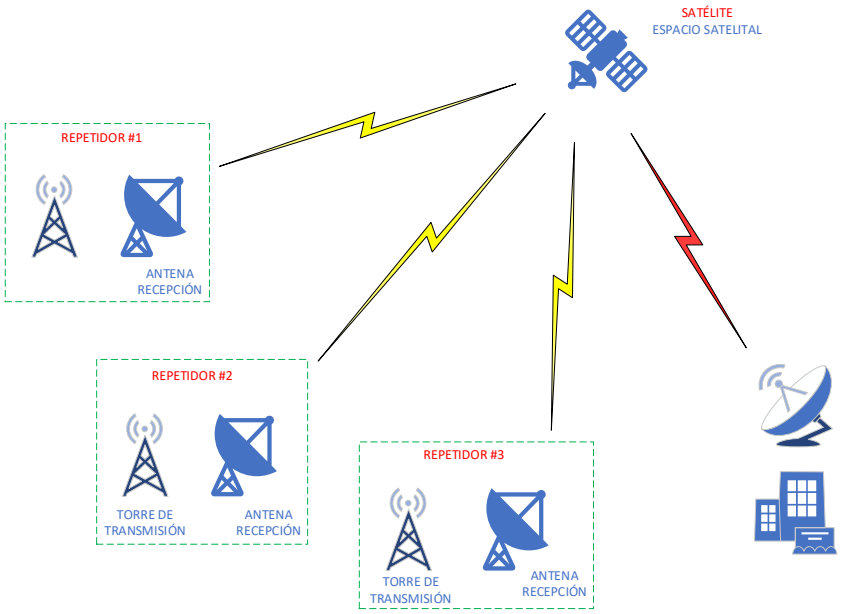

Figura 8.- Topología de una red de Radiodifusión o televisión, (Elaboración propia).

Los sitios donde se instalan los repetidores, son cerros existentes que de acuerdo con estudios técnicos se determina el más idóneo para instalar la infraestructura de la repetidora, el sitio debe garantizan tener la mejor cobertura posible para cubrir la mayor parte de zonas de sombra que puedan existir. Se puede mostrar en la tabla 2 algunos de los cerros existentes dentro del territorio ecuatoriano con sus respectivas coordenadas y en las ciudades en las que se encuentran ubicados.

NOMBRE	LATITUD	LONGITUD	Ubicación		Hsnm
			PROVINCIA	CANTÓN	(m)
AGUA SANTA	01°41'46.80"S	78°39'59.80"W	CHIMBORAZO	RIOBAMBA	3006
BARABON	02°53'34.40"S	79°05'15.10"W	AZUAY	CUENCA	3351
BIBLIAN (SAGRADO CORAZON)	02°43'07.70"S	78°52'53.40"W	CAÑAR	BIBLIAN	2878
BILOVAN	01°48'09.10"S	79°06'17.20"W	BOLIVAR	SAN MIGUEL	2662
CERRO ABITAGUA	01°24'53.90"S	78°08'31.00"W	PASTAZA	MERA	2250
CERRO ACHAYANDI	00°57'07.70"S	78°55'32.90"W	COTOPAXI	PUJILI	4102
CERRO ALTAR URCU	02°28'48.90"S	78°59'36.30"W	CAÑAR	CAÑAR	3215
CERRO AMULA RAYOLOMA	01°41'22.10"S	78°43'45.70"W	CHIMBORAZO	COLTA	3485
CERRO ANIMAS	02°28'28.20"S	80°28'03.40"W	GUAYAS	GUAYAQUIL	427
CERRO APAGUA	00°58'32.00"S	78°56'58.00"W	COTOPAXI	PUJILI	4032
CERRO AZUL	02°09'57.40"S	79°57'24.80"W	GUAYAS	GUAYAQUIL	324
CERRO AZUL ALTO	02°09'05.10"S	79°59'01.80"W	GUAYAS	GUAYAQUIL	398
CERRO BALAO	02°45'58.90"S	79°39'52.90"W	GUAYAS	NARANJAL	490
CERRO BALAO (ESMERALDAS)	00°57'41.30"N	79°41'25.70"W	ESMERALDAS	ESMERALDAS	192
CERRO BELLAVISTA	02°10'57.10"S	79°54'51.30"W	GUAYAS	GUAYAQUIL	100
CERRO BERMEJO	00°08'09.10"N	77°19'39.90"W	SUCUMBIOS	CASCALES	994
CERRO BIJAGUAL	00°39'28.70"S	79°17'41.00"W	SANTO DOMING	SANTO DOMINGO	798
CERRO BLANCO	00°12'33.70"N	78°20'16.40"W	IMBABURA	OTAVALO	3545
CERRO BOMBOLI	00°14'48.20"S	79°11'31.30"W	SANTO DOMING	SANTO DOMINGO	636
CERRO BOSCO	03°00'01.00"S	78°30'34.90"W	MORONA SANT.	LIMON INDANZA	2404
CERRO BUERAN	02°36'31.10"S	78°55'50.90"W	CAÑAR	BIBLIAN	3816
CERRO BUERAN (Tierras Negras)	02°35'59.40"S	78°55'44.40"W	CAÑAR	BIBLIAN	3828
CERRO CAHUAZHUN	02°52'33.40"S	78°49'01.60"W	AZUAY	GUALACEO	2999
CERRO CALVARIO	01°31'14.30"S	77°54'28.50"W	PASTAZA	PASTAZA	1171
CERRO CALVARIO(LOJA)	04°02'47.00"S	79°39'05.80"W	LOJA	PALTAS	1935
CERRO CAPADIA ALTO	01°25'24.20"S	78°57'01.10"W	BOLIVAR	GUARANDA	4460
CERRO CAPADIA CHICO	01°25'54.10"S	78°56'19.70"W	BOLIVAR	GUARANDA	4384
CERRO CAPAES	02°12'32.40"S	80°51'47.50"W	SANTA ELENA	SANTA ELENA	102
CERRO CARSHAO	02°26'23.20"S	78°57'03.80"W	CAÑAR	CAÑAR	4017
CERRO CAYAMBE	00°03'58.00"N	77°59'25.80"W	PICHINCHA	CAYAMBE	4216
CERRO CHIGUILPE ALTO	00°17'44.70"S	79°05'12.10"W	SANTO DOMING	SANTO DOMINGO	1172
CERRO CHIGUILPE BAJO	00°16'51.30"S	79°05'26.40"W	SANTO DOMING	SANTO DOMINGO	801
CERRO CHILLA	03°29'09.70"S	79°36'04.70"W	EL ORO	CHILLA	3450
CERRO CHILOLA	03°30'00.90"S	79°37'52.50"W	EL ORO	CHILLA	3596
CERRO COCHAPAMBA	02°47'41.10"S	79°25'39.10"W	AZUAY	CUENCA	3714
CERRO COJITAMBO	02°45'40.70"S	78°53'16.40"W	CAÑAR	AZOGUES	3106
CERRO COLAMBO	04°14'14.60"S	79°23'45.40"W	LOJA	GONZANAMA	3103
CERRO CONDIJUA	00°29'01.60"S	77°54'06.90"W	NAPO	QUIJOS	2526
CERRO COROZO	01°29'23.10"S	80°31'27.00"W	MANABI	JIPIJAPA	774
CERRO COTACACHI	00°19'57.30"N	78°20'24.60"W	IMBABURA	COTACACHI	4047
CERRO CRUZ LOMA	00°11'17.50"S	78°32'06.70"W	PICHINCHA	QUITO	3990
CERRO DE HOJAS	01°02'40.10"S	80°32'38.50"W	MANABI	PORTOVIEJO	674
CERRO DE LA COCHA	01°50'38.90"S	79°09'52.80"W	BOLIVAR	SAN MIGUEL	2496
CERRO DON JUAN	00°50'51.20"N	79°53'52.40"W	ESMERALDAS	ATACAMES	262
CERRO EL CARMEN	02°10'47.60"S	79°52'50.00"W	GUAYAS	GUAYAQUIL	80
CERRO EL TRIGAL	03°41'23.20"S	79°39'33.90"W	EL ORO	PIÑAS	1292
CERRO GATAZO	00°56'38.40"N	79°39'29.70"W	ESMERALDAS	ESMERALDAS	271
CERRO GATAZO GRANDE	01°39'41.30"S	78°45'00.80"W	CHIMBORAZO	COLTA	3188
CERRO GONZALEZ	02°20'57.30"S	80°32'36.70"W	GUAYAS	GUAYAQUIL	296
CERRO GUACHAURCO	04°02'11.70"S	79°52'19.40"W	LOJA	PALTAS	3102
CERRO GUAGUALZHUMI	02°53'32.50"S	78°54'39.60"W	AZUAY	CUENCA	3079
CERRO GUALLIL	03°04'30.90"S	78°48'58.70"W	AZUAY	SIGSIG	3284
CERRO GUANCHURO	04°04'28.80"S	79°38'54.90"W	LOJA	PALTAS	2446
CERRO GUAYAS	00°41'18.10"S	80°05'27.90"W	MANABI	CHONE	128
CERRO GUIRILLIO	03°09'57.50"S	79°36'20.40"W	EL ORO	EL GUABO	3192
CERRO HIGNO CACHA	01°41'31.90"S	78°42'58.60"W	CHIMBORAZO	RIOBAMBA	3567
CERRO HITO CRUZ	02°55'51.50"S	78°59'51.70"W	AZUAY	CUENCA	2847
CERRO HUACAMAYOS	00°37'26.06"S	77°50'28.60"W	NAPO	ARCHIDONA	2300
CERRO HUANCHINCHAMBO	04°01'25.20"S	79°14'45.90"W	LOJA	CATAMAYO	2849
CERRO ILAMBULO	01°42'18.60"S	79°05'32.60"W	BOLIVAR	SAN MIGUEL	3213
CERRO ILUMBISI	00°13'40.60"S	78°28'25.70"W	PICHINCHA	QUITO	3040
CERRO JAMA 1	00°12'29.40"S	80°16'08.70"W	MANABI	JAMA	93
CERRO JAMA 2	00°12'58.90"S	80°16'44.50"W	MANABI	JAMA	140
CERRO LA CHUVA	03°43'03.00"S	79°39'11.30"W	EL ORO	PIÑAS	1282

CERRO LA CRESPA	00°20'07.10"S	79°45'34.70"W	MANABI	FLAVIO ALFARO	480
CERRO LA JUANITA	00°28'02.20"N	79°34'40.50"W	ESMERALDAS	QUININDE	502
CERRO LA MIRA	01°30'32.70"S	78°35'05.20"W	CHIMBORAZO	GUANO	3881
CERRO LA MIRA (SENINCAHUA)	01°41'31.90"S	78°42'58.60"W	CHIMBORAZO	RIOBAMBA	3947
CERRO LA VIRGEN	00°19'06.90"S	78°11'27.60"W	PICHINCHA	QUITO	4412
CERRO LUMBAQUI ALTO	00°00'30.40"N	77°19'15.80"W	ORELLANA	ORELLANA	1088
CERRO LUMBAQUI BAJO	00°01'47.70"N	77°19'07.90"W	ORELLANA	ORELLANA	998
CERRO MAPASINGUE	02°08'49.80"S	79°55'40.80"W	GUAYAS	GUAYAQUIL	120
CERRO MOTILON	04°04'57.10"S	79°56'22.20"W	LOJA	CELICA	2663
CERRO NITON	01°16'41.60"S	78°32'10.20"W	TUNGURAHUA	San Pedro Pelileo	3079
CERRO NUEVE	00°16'05.30"S	80°12'29.90"W	MANABI	JAMA	652
CERRO PILISURCO o SAGUATOA	01°09'17.20"S	78°39'58.00"W	TUNGURAHUA	AMBATO	4154
CERRO PUCARA	01°39'04.50"S	79°06'10.70"W	BOLIVAR	CHIMBO	3063
CERRO PUENGASI	00°14'43.40"S	78°29'59.70"W	PICHINCHA	QUITO	3085
CERRO PUERTO ALTO	00°21'36.30"S	79°52'00.40"W	MANABI	FLAVIO ALFARO	410
CERRO PUTZALAGUA	00°57'56.00"S	78°33'42.70"W	COTOPAXI	LATACUNGA	3531
CERRO QUILAMO	02°18'14.40"S	78°08'26.80"W	MORONA SANT.	MORONA	1461
CERRO RAYOLOMA	02°54'13.00"S	78°57'59.20"W	AZUAY	CUENCA	2649
CERRO REPEN	03°33'01.80"S	79°41'02.70"W	EL ORO	ATAHUALPA	2369
CERRO REVENTADOR	00°02'33.90"S	77°31'44.10"W	SUCUMBIOS	GONZALO PIZARR	1600
CERRO RUMILOMA (AMPUNGO)	02°48'19.00"S	78°49'19.30"W	AZUAY	PAUTE	3175
CERRO SALVACION (LLIGUA)	01°22'44.90"S	78°26'11.80"W	TUNGURAHUA	BAÑOS	2696
CERRO SAN EDUARDO	02°10'28.00"S	79°55'07.30"W	GUAYAS	GUAYAQUIL	160
CERRO SAN LUIS	00°35'10.70"S	79°19'18.80"W	SANTO DOMING	SANTO DOMINGO	412
CERRO SANTA ANA	01°55'45.30"S	79°45'50.10"W	GUAYAS	SAMBORONDON	300
CERRO TANZARTE (CRUZ LOMA)	02°45'17.90"S	79°23'59.80"W	GUAYAS	NARANJAL	2894
CERRO TRES CRUCES	00°16'22.60"S	77°45'53.80"W	NAPO	EL CHACO	1980
CERRO TROYA	00°44'24.50"N	77°41'48.70"W	CARCHI	TULCAN	3529
CERRO TROYA BAJO	00°44'09.50"N	77°42'22.80"W	CARCHI	TULCAN	3418
CERRO TURI	02°55'22.90"S	79°00'32.80"W	AZUAY	CUENCA	2730
CERRO VENTANAS	04°01'54.10"S	79°14'39.00"W	LOJA	CATAMAYO	2870
CERRO VILLAFLOR	02°49'44.60"S	78°48'12.30"W	CAÑAR	AZOGUES	3042
CERRO VILLONACO ALTO	03°59'18.70"S	79°16'06.30"W	LOJA	CATAMAYO	2952
CERRO ZAPALLO	00°53'05.30"N	79°31'52.20"W	ESMERALDAS	ESMERALDAS	690
CHASQUI	00°37'15.30"S	78°35'37.20"W	COTOPAXI	LATACUNGA	3532
COCHABAMBA (CHUCHI)	01°41'56.20"S	79°06'20.60"W	BOLIVAR	SAN MIGUEL	3072
LLANTANTOMA	01°09'56.40"S	78°37'53.90"W	TUNGURAHUA	AMBATO	3450
LOMA DE LOURDES	01°42'19.70"S	79°04'49.20"W	BOLIVAR	SAN MIGUEL	3281
LOMA DE VIENTO	00°42'28.10"S	80°24'28.80"W	MANABI	SUCRE	402
MIRADOR BAHIA DE CARAQUEZ	00°36'28.00"S	80°25'34.70"W	MANABI	SUCRE	102
MOJANDA	00°04'47.30"N	78°13'43.20"W	PICHINCHA	PEDRO MONCAY	3241
PADRE URCU	01°42'21.80"S	79°05'45.70"W	BOLIVAR	SAN MIGUEL	3181
PAPALLACTA	00°22'43.70"S	78°08'48.60"W	NAPO	QUIJOS	3373
PUCARA	04°05'47.80"S	79°56'17.70"W	LOJA	CELICA	2479
SANTA CLARA (Km.35 vía Puyo-T)	01°17'29.30"S	77°52'36.80"W	PASTAZA	SANTA CLARA	1080
SARAGURO	03°37'55.20"S	79°14'37.50"W	LOJA	SARAGURO	2740
SECTOR FERROVIARIA/FOREST	00°15'48.00"S	78°30'25.00"W	PICHICHA	QUITO	3179
TIPOLOMA ALTO	03°08'13.20"S	79°05'09.70"W	AZUAY	GIRON	3177
TIPOLOMA BAJO	03°07'41.50"S	79°05'17.90"W	AZUAY	GIRON	3086
CERRO CABRAS	00°28'15.00"N	77°57'50.70" W	CARCHI	BOLIVAR	2904
CERRO CHILES ALTO	00°35'14.40"N	77°51'04.00"W	CARCHI	MONTUFAR	3093
CERRO CONDORCOCHA	00°02'19.10"S	78°30'41.00"W	PICHINCHA	QUITO	3586
CERRO PAREDONES	02°44'38.30"S	79°26'26.60"W	AZUAY	CUENCA	3668
CUCHILLA DE PAREDONES	02°44'10.70"S	79°26'43.10"W	AZUAY	CUENCA	3311
CERRO SAN FRANCISCO	00°23'24.89"S	78°37'06.79"W	PICHINCHA	MEJIA	4097
CERRO TINAJERO	00°22'53.08"S	78°36'36.19"W	PICHINCHA	MEJIA	4082
CERRO LA VIUDITA	00°24'50.70"S	78°36'27.00"W	PICHINCHA	MEJIA	3751
CERRO HACDA EL ROSARIO	00°26'14.60"S	78°32'12.50"W	PICHINCHA	MEJIA	2981
CERRO ATACAZO ALTO	00°21'22.20"S	78°37'09.10"W	PICHINCHA	QUITO	4474
CERRO ATACAZO BAJO	00°19'05.10"S	78°36'08.10"W	PICHINCHA	QUITO	3893
CERRO PICHINCHA	00°09'57.03"S	78°31'39.19"W	PICHINCHA	QUITO	3901

Tabla 2. Lista de sitios con sus coordenadas de referencia, (Paredes, 2013).

La base de datos de la ubicación de cerros descritos en la tabla 2, muestra claramente sus coordenadas y la ubicación en las respectivas provincias y ciudades en las que se encuentran a nivel nacional; también se puede ver la altura a la que se encuentra el cerro tomando como referencia sobre el nivel del mar, este tipo de datos es importante cuando se va a realizar un estudio técnico para la solicitud del concurso de frecuencias.

Se debe considerar de mucha importancia el camino de acceso hacia un cerro y lo complejo que resulta viajas hacia el sitio, tomando en cuenta varios factores que se pueden producir al momento de realizar una atención en una estación que se encuentre fuera del aire. Así también se tomará en cuenta el sitio donde se puede instalar el sistema de monitoreo y control para facilitar una mejor atención, optimizando tiempo y recursos de una empresa.

El camino de acceso a las repetidoras que se encuentran en ciudades pequeñas es complejo, el camino puede encontrarse en buen estado como puede estar en mal estado; en la mayor parte de cerros el camino no es bueno para acceder con vehículo normal. Los caminos pueden ser de tipo empedrado en parte y puramente de tierra, e inclusive puede llegar a ser pantanoso debido a las condiciones climáticas que exista en el cerro.

Debido a las causas presentadas se puede mostrar en las imágenes de la figura 9, el estado del camino para acceder a una estación de radio. Se puede observar que el camino debido a las condiciones climáticas es pantanoso y solo se puede acceder con vehículos que tengan la funcionalidad de 4X4 incorporada; aunque existe ocasiones en que el camino no favorece para ningún tipo de vehículos y solo se puede acceder a pie, lo cual demora la llegada a una estación que se encuentre con fallas.

Figura 9. Fotos de camino de acceso hacia un cerro, (Foto autoría propia).

También se debe considerar el tiempo que demora trasladarse al sitio donde se va a brindar atención, ya que el cerro que se ha tomado como referencia se llama “Los libres” ubicado dentro de la provincia Santo Domingo de los Tsachilas, en la vía Aloag-Santo Domingo. Solo hasta llegar desde Quito hasta la entrada que se encuentra marcado con una X en la figura 10, toma un tiempo promedio de 2 horas y adicional el tiempo por la carretera en mal estado desde la entrada hasta llegar a la estación repetidora toma 1 Hora. Este es un ejemplo de tiempos que se maneja en trasladarse a una repetidora para dar una atención, así también se debe considerar que puede existir cierre de vías por derrumbes u otros problemas que se pueden ocasionar durante el traslado hasta el sitio.

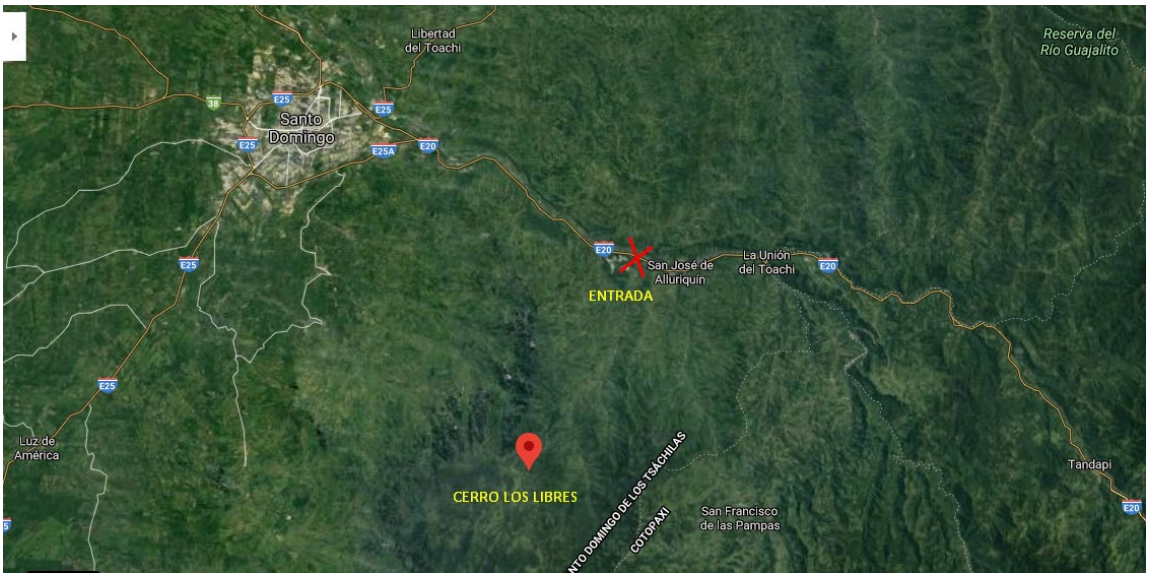

Figura 10.- Mapa de ubicación cerro Los Libres, (Gráfico de Google Maps).

Una vez descrito las condiciones del camino de acceso se puede presentar fotografías del interior de una estación, para tener una idea de qué tipo de equipos se puede tener internamente como parte de un sistema de Radio y/o televisión. En las imágenes de la figura 11; se muestra los equipos que básicamente operan en una estación repetidora, la foto del lado izquierdo corresponde a equipos que trabajan para el servicio de Televisión, en la parte baja se tiene el transmisor de TV y en la parte superior equipos de Tx y Rx de enlace para recibir la señal de programa, que a diferencia de la topología planteada en la figura 8; este sistema se conecta a la estación repetidora por enlaces. En la imagen derecha se observa un sistema para el servicio de Radio, que de la misma manera para obtener el servicio en la estación se lo hace a través de equipos de enlace de radio mismo que se observa en la parte baja del rack y en la parte superior se encuentra el transmisor.

Figura 11. Sistemas de Radio y Televisión en una estación, (Foto autoría propia).

De las imágenes mostradas en la figura 11, se puede mencionar que estos son sistemas que se encuentran en estaciones repetidoras remotas que dan los servicios de Televisión y Radio respectivamente, y la cobertura que cubre son para ciudades pequeñas y/o pueblos, en la imagen de la figura 12 se muestra el sistema Radiante que se instala en la torre para poder transmitir una potencia generada por los diferentes equipos de transmisión.

Figura 12.- Sistema radiante montado en la torre, (Foto autoría propia).

Todo lo detallado anteriormente, muestra las condiciones en las que se puede encontrar un sitio donde se instala una repetidora; así como los equipos que básicamente se pueden encontrar en la estación y el tiempo que dura la movilización, pudiendo variar por las

condiciones que se pueden dar durante la movilización hasta el lugar donde se va a realizar una atención de mantenimiento correctivo.

2.2. Internet de las cosas

El Internet de las cosas es un tipo de tecnología que actualmente se encuentra evolucionando y que gracias a su gran variedad de aplicaciones que existe en la actualidad, se puede aplicar en diferentes campos de la industria para optimizar recursos y mejorar la productividad.

2.2.1 ¿Qué es el Internet de las Cosas IoT?

El Internet de las cosas (IoT) se puede definir como una tecnología que actualmente se encuentra en evolución, esta tecnología permite la interconexión de objetos que se usa en la vida diaria hacia la red global del Internet para poder monitorearlos mediante la recolección de información de varios tipos de sensores y controlarlos mediante la creación de APIs que permiten la interacción con toda la red de objetos conectados en los diferentes campos de aplicación.

Cuando se habla de cosas, se refiere a objetos que se usa a diario y que son capaces de conectarse a Internet como los smartphones, computadoras, Smart TV y video juegos, la idea de la tecnología es tener un dispositivo que sean más eficientes y que puedan ser capaces de recoger más información útil, por ejemplo se puede hablar de dispositivos que contribuyan a optimizar recursos naturales, parte de la salud y otras innumerables opciones que se puedan presentar (Valois, 2018).

Se puede mostrar en la figura 13, una imagen acerca de la interconexión de dispositivos como la televisión, un foco, el auto, celular, laptop, refrigeradora, cafetera, etc. Son objetos que se usa en la vida diaria y que para controlarlos se conectan hacia la red de Internet con el fin de poder monitorearlos desde cualquier parte del planeta.

Gracias a la tecnología de redes inalámbricas y el bajo costo de nuevos procesadores, hoy en día es posible casi cualquier cosa, desde una aspiradora inteligente hasta un vehículo autónomo que forma parte del Internet de las cosas. Todo esto permite agregar un nivel de inteligencia a dispositivos que son capaces de comunicar datos en tiempo real sin tener que estar un ser humano presente (Peña, 2019).

Figura 13.- Interconexión de objetos hacia el Internet, *(ComoFuncionaQue, 2020)*.

2.2.2 ¿Como funciona?

El envío masivo de información procedente de dispositivos integrados, cambia radicalmente las actividades que se hace en la vida diaria como: el cocinar, trabajar, vestirse, caminar, correr, etc. Por lo que todos son influidos en mayor o menor medida.

Como se puede observar en la figura 14, los procesos mostrados identifican el funcionamiento de la tecnología, para tener una idea clara de su funcionamiento.

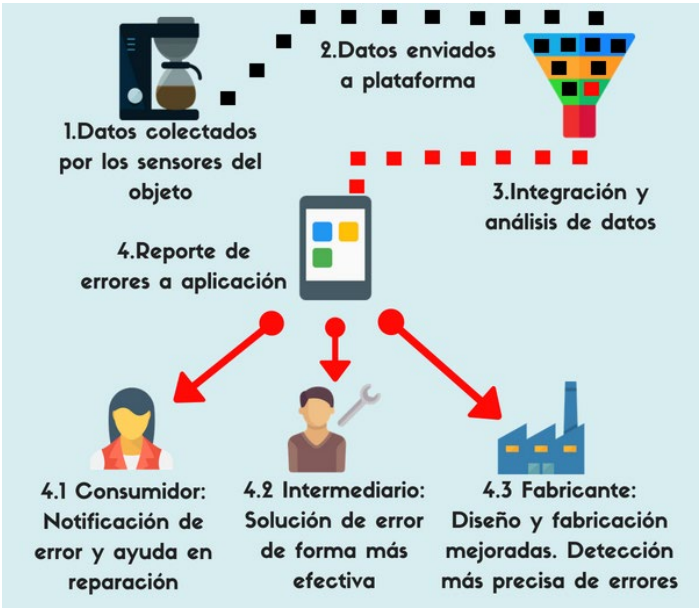

Figura 14. ¿Cómo funciona IoT?, *(Pocest, 2019)*.

Cada elemento tendrá una serie de sensores de presión, temperatura, etc.… Estos sensores, envían la información al dispositivo inmerso en el propio elemento para que la procese, y posteriormente se envía a la red para llegar a su siguiente destino. Una plataforma colectora de datos (Pocest, 2019).

Esta plataforma recibe datos de múltiples dispositivos, los integra para quedarse con la información más importante y posteriormente realiza un análisis exhaustivo. Una vez finalizado el análisis, la información se envía a una aplicación (Pocest, 2019).

Esta aplicación, permite ver a partir de los datos cuando hay riesgo de error. Lo usan tres tipos de usuarios:

Consumidor: una vez que se detecte un problema real, se puede notificar al usuario y se le dan recomendaciones de cuál es la mejor forma de arreglarlo (Pocest, 2019).

Intermediario: Una vez el consumidor haya confirmado que quiere resolver el problema, la información en detalle llegará al grupo de trabajo encargado de resolverlo. Esto permitirá que se trabaje con antelación en la solución del mismo, al no tener que disponer del producto directamente para todo el proceso (Pocest, 2019).

Fabricante: Toda la información recopilada se usará para mejorar el diseño y fabricación del producto. También permite detectar fallos en sus procesos, incluyendo exactamente donde y cuando ocurrieron, ya que cada componente de un producto tiene un registro electrónico asociado (Pocest, 2019).

Sin duda, el principal beneficio del Internet de las cosas es la prevención. Al poder detectar múltiples indicadores constantemente, te anticipas al problema mucho más que en la actualidad. Aunque por supuesto, el proceso de arreglo y de mejora tampoco se quedarán atrás (Pocest, 2019).

2.2.3 Protocolos utilizados en IoT

Entre los protocolos de comunicaciones que usa el Internet de las Cosas, se puede encontrar varios, entre los que se destacan MQTT y CoAP; el resto de protocolos se deben estudiar si se requiere una aplicación especifica. Los dos protocolos mencionados son los que se van a estudiar en este subtema del capítulo, para explicar el funcionamiento de cada protocolo.

2.2.3.1 *MQTT (Message Queue Telemetry Transport)*

El protocolo MQTT es un tipo de protocolo usado para la comunicación Maquina a Maquina (M2M), se utiliza para la comunicación entre dispositivos catalogados como IoT y que están conectados en una red. Dentro de la tecnología de IoT el protocolo MQTT, se puede tratar como un pilar importante debido a su sencillez y ligereza; estos dos factores condicionan para mantener limitaciones de potencia, ancho de banda y consumo.

El protocolo MQTT se basa en la pila TCP/IP como base para una buena comunicación. MQTT mantiene abierta cada comunicación y se reutiliza en cada comunicación, a diferencia de una petición HTTP 1.0 que para cada transmisión se origina una conexión (Llamas, 2019).

2.2.3.1.1. *FUNCIONAMIENTO MQTT*

Para el funcionamiento se considera como un servicio de mensajería PUSH con patrón publicador/suscriptor (pub-sub), que indica que cada cliente que se conecta debe hacerlo a un servidor central llamado broker. Los mensajes que se envían a los clientes se filtran mediante topics (temas) que se organizan jerárquicamente para poder acceder a la información, cada cliente puede suscribirse a un topic para poder acceder a la información únicamente de ese topic, el bróker es el encargado de hacer llegar los mensajes a los clientes.

En la figura 15 se muestra el diagrama de cómo se publica la información y mediante el bróker quien filtra, envía esta información hacia un cliente quien previamente se subscribe a un topic.

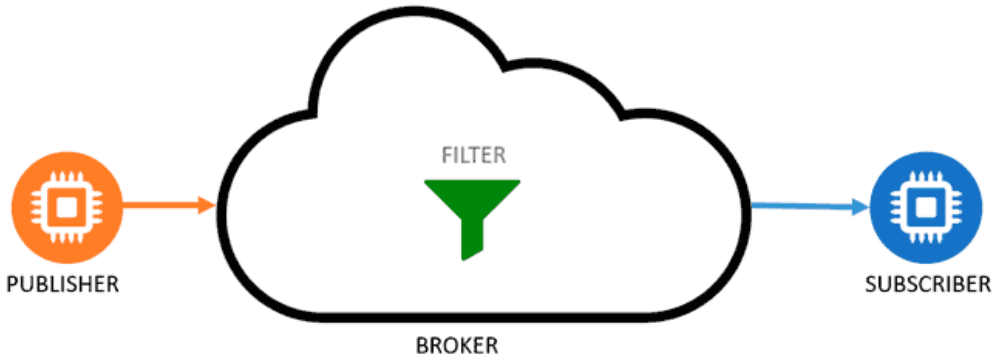

Figura 15. Publicación y Subscripción de mensaje, *(Llamas, 2019)*.

Un cliente debe iniciar una conexión TCP/IP con el broker central, donde se mantiene el registro de todos los clientes conectados. Cualquier conexión abierta se mantendrá para poder reusarla hasta que el mismo cliente finalice. Para la comunicación por defecto el protocolo MQTT utiliza el puerto 1883 y el puerto 8883 cuando trabaja con seguridad TLS (Transport Layer Security) (Llamas, 2019).

En la figura 16 se observa la conexión del cliente, para iniciar la conexión el cliente envía un mensaje CONNECT donde viaja la información necesaria como nombre de usuario, contraseña, ID del cliente, etc., el broker central debe responder un mensaje CONNACK que contendrá la aceptación o rechazo de la conexión (Llamas, 2019).

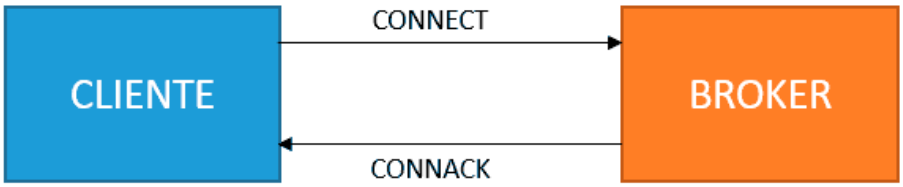

Figura 16. Proceso de conexión, *(Llamas, 2019)*.

Una vez aceptada la conexión, el cliente envía mensajes de PUBLISH, en los mensajes se envía información del topic y el payload.

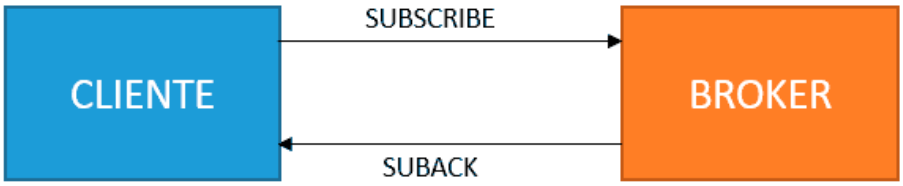

Figura 17. Envió mensaje de Publish, *(Llamas, 2019)*.

En la figura 17 se muestra el proceso de suscripción y eliminación de suscripción, para lo cual se emplean mensaje de SUBSCRIBE y UNSUBSCRIBE, como respuesta a los mensajes el broker responde con mensajes de SUBACK y UNSUBACK.

Para garantizar la continuidad de la conexión; es decir, que la conexión este activa el cliente o los clientes deben enviar cada cierto tiempo un mensaje de PINGREQ que debe ser respondido por el broker con un PINGRESP. Y para que el cliente se desconecte se debe enviar un mensaje de DISCONNECT.

2.2.3.1.2. *ESTRUCTURA*

La estructura de un mensaje MQTT se muestra en la figura 18, cada mensaje enviado se forma de tres partes: cabecera fija, cabecera variable y payload.

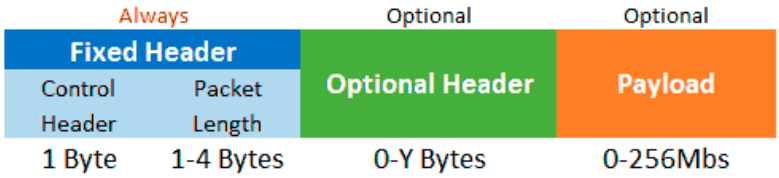

Figura 18. Estructura de un mensaje MQTT, *(Llamas, 2019)*.

Cabecera fija. – la cabecera fija tiene longitud de 2 a 5 bytes, obligatorio. Contiene un código de control para identificar el tipo de mensaje que se ha enviado y la longitud, la longitud del mensaje se codifica de 1 a 4 bytes, dentro de los cuales se usan los 7 primeros bits y el ultimo bit como uno de continuidad (Llamas, 2019).

Cabecera variable. – es un campo opcional que contiene información que puede ser necesaria en ciertas situaciones o mensajes (Llamas, 2019).

Contenido (payload). – la longitud del mensaje puede ser máximo de 256 Mb, sin embargo, en implementaciones reales el máximo puede ser de 2 a 4 Kb (Llamas, 2019).

La tabla 3 contiene los tipos códigos de control y mensajes que se pueden enviar dentro del protocolo MQTT.

MENSAJE	CODIGO
CONNECT	0x10
CONNACK	0x20
PUBLISH	0x30
PUBACK	0x40
PUBREC	0x50
PUBREL	0x60
PUBCOM	0x70
SUSBSCRBE	0x80
SUBACK	0x90
UNSUSCRIBE	0xA0
UNSUBACK	0xB0
PINGREQ	0xC=
PINGRESP	0xD0
DISCONNECT	0xE0

Tabla 3.- Códigos de Control y mensajes en protocolo MQTT, *(Llamas, 2019)*.

2.2.3.1.3. *CALIDAD DE SERVICIO QoS*

El protocolo MQTT posee un método de QoS, que se basa en la forma de gestionar la robustez que posee un mensaje cuando se envía a un cliente, ante fallas que se producen durante él envió como por ejemplo fallas de conectividad. Cuando se habla de calidad de servicio en MQTT, se puede definir tres niveles posibles de QoS.

QoS 0 unacknowledged (como máximo uno): este nivel de calidad de servicio indica que el mensaje MQTT se envía una sola vez y en caso de errores no se garantiza la entrega.

QoS 1 acknowledged (al menos uno): este nivel de QoS garantiza la entrega del mensaje MQTT, ya que se envían hasta entregarse, por lo que el suscriptor puede recibir mensajes duplicados.

QoS 2 assured (exactamente uno): este nivel garantiza que el mensaje MQTT llegue al suscriptor una única vez.

La utilización de cualquiera de estos niveles de QoS depende de las necesidades y requerimientos de fiabilidad que se desee; sin embargo, se debe tomar en cuenta que un nivel de calidad de servicio mayor, necesita intercambiar mayor número de mensajes con el cliente y como consecuencia se tendrá mayor carga en el sistema y requerirá mayor ancho de banda en la transmisión.

2.2.3.1.4. *SEGURIDAD EN MQTT*

La seguridad es u8n factor importante que se debe considerar en los sistemas de comunicación M2M, por este motivo el protocolo permite distintos modos de seguridad que ayuda a proteger las comunicaciones (Llamas, 2019).

Los tipos de seguridad que se tiene incluye transporte SSL/TLS y autentificación por usuario y contraseña o mediante certificado, el protocolo como tal permite estos tipos de seguridad; pero, se debe tener en cuenta que los dispositivos usados para IoT tienen escasa capacidad y la seguridad SSL/TLS puede suponer una carga de proceso importante (Llamas, 2019).

Lo mencionado anteriormente de considerarse cuando se configura un sistema con protocolos MQTT, tomando en cuenta los riesgos y las consecuencias que puede afectar a la eficiencia del sistema.

VENTAJAS DEL MQTT

- El protocolo MQTT es sencillo y ligero, estas características hacen que se adecuado en aplicaciones de IoT que emplean dispositivos de baja potencia.
- El uso de menores recursos implica menor consumo de energía, lo que hace favorable cuando se trabaja con dispositivos que funcionan 24/7 y que trabajan con baterías.
- La ligereza del protocolo MQTT hace que el ancho de banda requerido sea menor, lo cual es una ventaja cuando se trabaja con redes inalámbricas.
- El protocolo MQTT posee seguridad y calidad de servicio, lo que hace que el protocolo sea robusto y confiable.

2.2.3.2 *CoAP (Constrained Application Protocol)*

Es un protocolo utilizado para el Internet de las Cosas IoT, significa Protocolo de Aplicación restringida, se define en el RFC7252. Es un protocolo de baja sobrecarga diseñado para dispositivos restringidos como microcontroladores, se utiliza para el intercambio de datos maquina a máquina (M2M) y es similar a HTTP.

El protocolo CoAP posee algunas características como:

- Protocolo web utilizado en M2M con requisitos restringidos.
- Intercambio asíncrono de mensajes.
- Bajo costo y fácil de analizar.
- URI y soporte de tipo de contenido.
- Proxy y capacidades de almacenamiento cache.

En la figura 19, se puede ver la representación de CoAP, existe dos capas que forman el protocolo: Menssages y Request/Response. La capa Messages trabaja con UDP y mensajes asíncronos mientras que la capa Request/Response gestiona la interacción de esta capa en función de los mensajes de Request/Response.

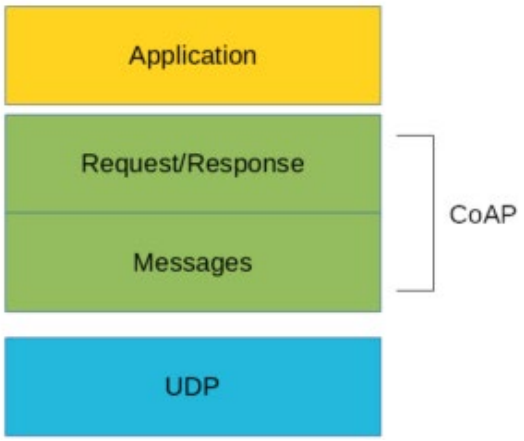

Figura 19. Representación de CoAP desde la capa de protocolo de abstracción, *(Azzola, 2018)*.

CoAP soporta cuatro diferentes tipos de mensaje: Confirmable, Non-confirmable, Acknowledgment y Reset.

2.2.3.2.1. *CAPA MESSAGE*

Esta es la capa más baja de CoAP. Esta capa trata con UDP intercambiando mensajes entre puntos finales. Cada mensaje de CoAP tiene una identificación única; Esto es útil para detectar mensajes duplicados. Las partes que forman un mensaje CoAP son: encabezado binario, opciones compactas, y carga útil (Azzola, 2018).

El protocolo CoAP usa dos tipos de mensajes: Mensaje confirmable y mensaje no confirmable. Un mensaje confirmable es un mensaje confiable. Al intercambiar mensajes entre dos puntos finales, estos mensajes pueden ser confiables. En CoAP, se obtiene un mensaje confiable usando un mensaje Confirmable (CON). Al usar este tipo de mensaje, el cliente puede estar seguro de que el mensaje llegará al servidor. Se envía un mensaje confirmable una y otra vez hasta que la otra parte envíe un mensaje de confirmación (ACK). El mensaje ACK contiene la misma ID del mensaje confirmable (CON) (Azzola, 2018). Este proceso de envío y confirmación se muestra en la figura 20.

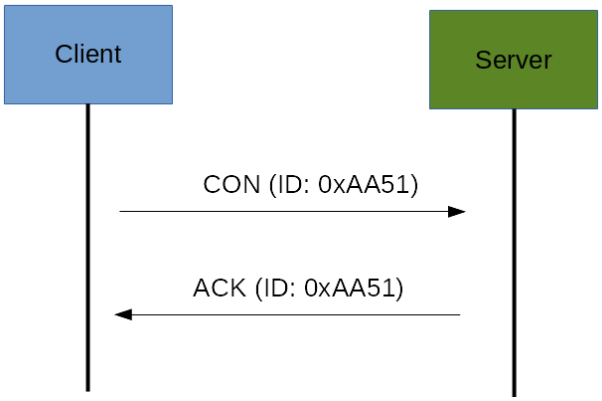

Figura 20. Proceso de envío y confirmación, *(Azzola, 2018)*.

Si el servidor tiene problemas para administrar la solicitud entrante, puede enviar un mensaje de descanso (RST) en lugar del mensaje de reconocimiento (ACK) (Azzola, 2018). Como se muestra en la figura 21.

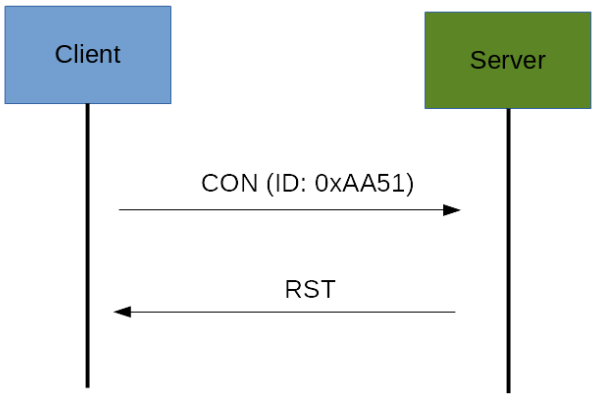

Figura 21. Envío mensaje RST cuando no se puede administrar el mensaje, *(Azzola, 2018)*.

La otra categoría de mensajes son los mensajes no confirmables (NO). Estos son mensajes que no requieren un reconocimiento por parte del servidor. Son mensajes poco confiables, es decir, mensajes que no contienen información crítica que debe entregarse al servidor. A esta categoría pertenecen los mensajes que contienen valores leídos de los sensores. Incluso si estos mensajes no son confiables, tienen una identificación única como se muestra en la figura 22.

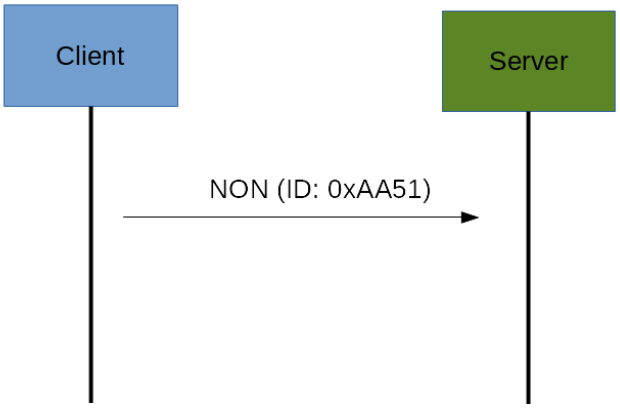

Figura 22. Envío mensajes no confirmables, *(Azzola, 2018)*.

2.2.3.2.2. *CAPA REQUEST / RESPONSE*

Es la capa alta de CoAP. La solicitud se envía mediante un mensaje confirmable (CON) o no confirmable (NO). Existen varios escenarios, la primera es si el servidor puede responder de inmediato la solicitud del cliente y la segunda si la respuesta no está disponible.

Cuando el servidor responde inmediatamente, la solicitud se lleva a cabo con un mensaje confirmable (CON), entonces se envía de vuelta al cliente un mensaje de reconocimiento con la respuesta o código de error como muestra la figura 23.

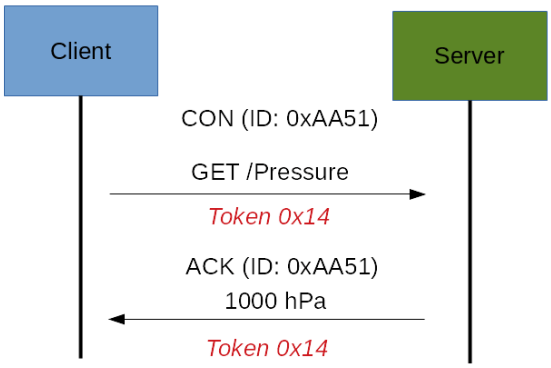

Figura 23.- Respuesta inmediata desde el servidor, *(Azzola, 2018)*.

En la figura 23 se observa que el token es diferente del ID de mensaje, esto se utiliza para hacer coincidir la solicitud y la respuesta.

Si el servidor no puede responder a la solicitud del cliente inmediatamente, se envía un mensaje de reconocimiento con una respuesta vacía. Y cuando la respuesta esté

disponible, el servidor envía un nuevo mensaje Confirmable al cliente con la respuesta. En este punto, el cliente devuelve un mensaje de confirmación. Este proceso se muestra en la figura 24.

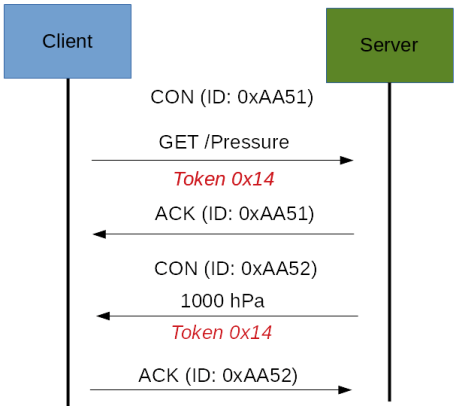

Figura 24. Respuesta no inmediata desde el servidor, *(Azzola, 2018)*.

Si la solicitud que proviene del cliente se lleva a cabo utilizando un mensaje NO confirmable, entonces el servidor responde utilizando un mensaje NO confirmable (Azzola, 2018).

2.2.3.2.3. *ESTRUCTURA*

El protocolo CoAP es la parte fundamental para entornos restringidos y, por esta razón, utiliza mensajes compactos. Para evitar la fragmentación, un mensaje ocupa la sección de datos de un datagrama UDP. La estructura del mensaje CoAP se muestra en la figura 25.

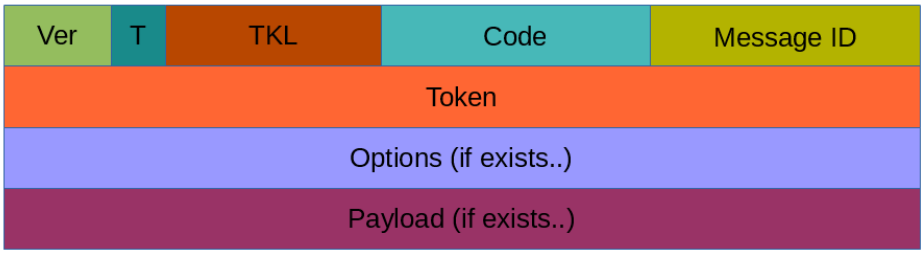

Figura 25. Estructura Protocolo CoAP, *(Azzola, 2018)*.

Donde:

Ver: es un entero sin signo de 2 bits que indica la versión

T: es un entero sin signo de 2 bits que indica el tipo de mensaje: 0 confirmable, 1 no confirmable

TKL: la longitud del token es la longitud del token de 4 bits

Código: es la respuesta del código (longitud de 8 bits)

ID del mensaje: es la ID del mensaje expresada con 16 bits

2.2.3.2.4. *ASPECTOS DE SEGURIDAD DE COAP*

CoAP usa UDP para transportar información. CoAP se basa en aspectos de seguridad UDP para proteger la información. Como HTTP usa TLS sobre TCP, CoAP usa Datagram TLS sobre UDP como se muestra en la figura 26. DTLS es compatible con RSA, AES, etc. De todos modos, se debe considerar que en algunos dispositivos restringidos algunos de los trajes de cifrado DTLS pueden no estar disponibles. Es importante tener en cuenta que algunas suites de cifrado introducen cierta complejidad y los dispositivos restringidos pueden no tener recursos suficientes para administrarlo (Azzola, 2018).

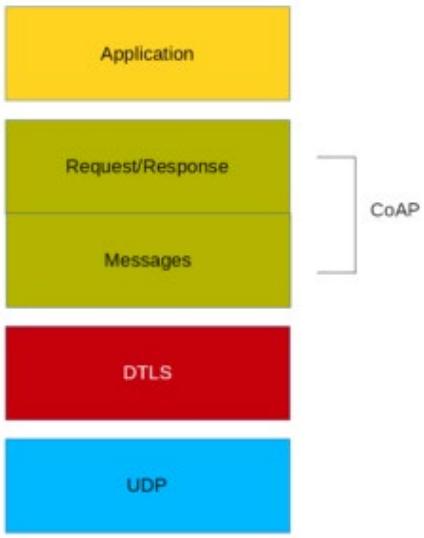

Figura 26. Seguridad de CoAP, *(Azzola, 2018)*.

2.2.3.3 *Comparación entre MQTT y CoAP*

Anteriormente se vio que MQTT es un protocolo de publicación-suscripción que facilita la comunicación entre un gran número de dispositivos y los gestiona mediante brokers. Los clientes publican los mensajes hacia un broker y/o se suscriben a un broker para recibir sus mensajes. Los mensajes están organizados por Topics (temas), qué básicamente son “etiquetas” que actúan como un sistema para enviar los mensajes a los suscriptores (Pick Data, 2019). Como se ve en la figura 27.

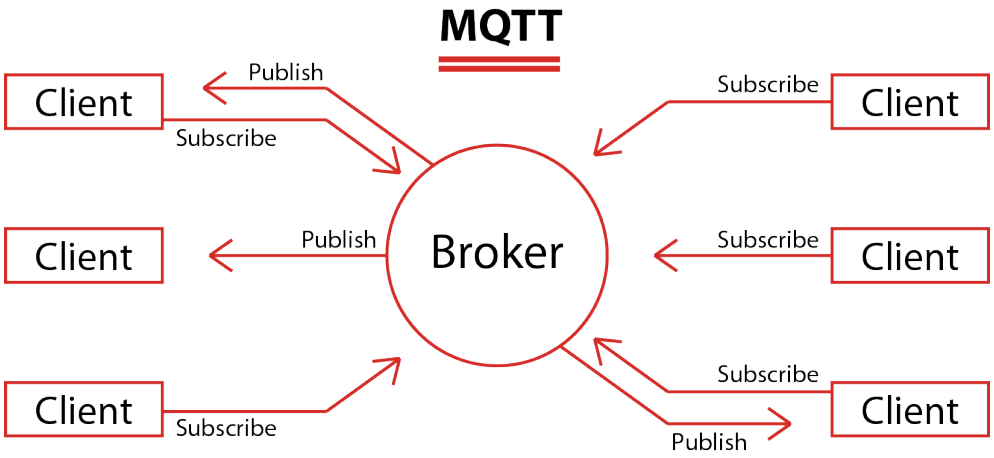

Figura 27. Publicación y suscripción de MQTT, *(Pick Data, 2019)*.

En cambio, CoAP es un protocolo cliente-servidor que no se encuentra estandarizado a un 100%. Con CoAP se puede tener que un nodo puede manejar a otro nodo mediante un paquete CoAP, el servidor CoAP interpreta y extrae la información del paquete, y toma una decisión a realizar dependiendo de su lógica, no es necesario que se realice una confirmación de petición (Pick Data, 2019). El esquemático de CoAP se puede mostrar en la Figura 28.

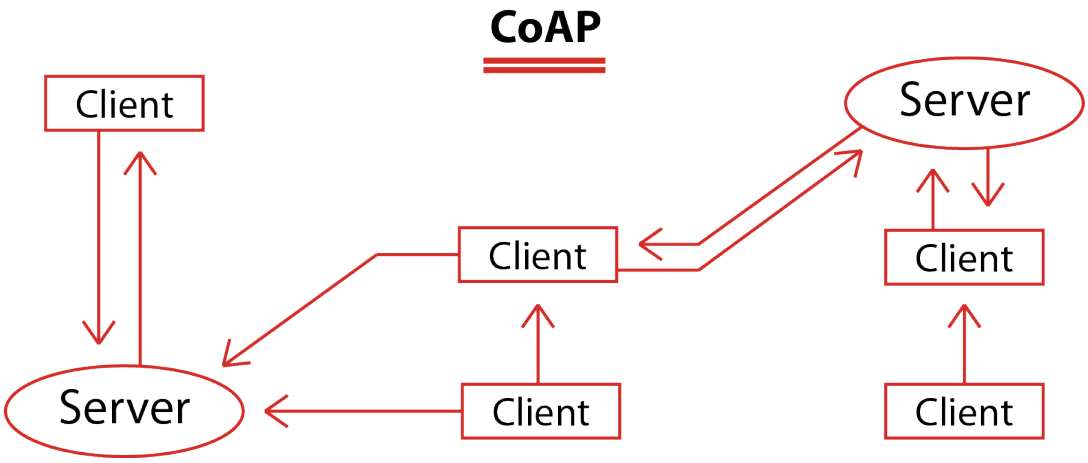

Figura 28.- CoAP trabaja como Cliente-Servidor, *(Pick Data, 2019)*.

Algunas de las diferencias de estos dos tipos de protocolos utilizados para IoT se muestra en la Tabla 4 a continuación:

CARACTERÍSTICAS	MQTT	COAP
Protocolo base	TCP	UDP
Modelo usado para la comunicación	Publicación-Suscripción	Pregunta-Respuesta Publicación-Suscripción
Nodo de comunicación	M: N	1:1
Consumo de energía	Mayor que CoAP	Menor que MQTT
RESTful	No	Sí
Número de tipos de mensaje usados	16	4
Tamaño de cabecera	2 bytes	4 bytes
Tipos de mensaje	Asíncrono	Asíncrono & Síncrono
Fiabilidad	Tres niveles de QoS: QoS 0: Entrega no garantizada QoS 1: Confirmación de entrega QoS 2: Doble confirmación de entrega	Mensajes confirmables Mensajes no confirmables ACKs Retransmisiones
Implementación	Fácil de implementar Complejo para añadir extensiones	Pocas librerías existentes y soporte
Seguridad	No definida Puede utilizar TLS/SSL	DTLS o IPSec
Otros	Útil para conexiones en localizaciones remotas Sin gestión del error	Baja saturación Baja latencia Problemas en NAT

Tabla 4.- Comparación entre MQTT y CoAP, *(Pick Data, 2019)*.

2.3. Soluciones de hardware y posibles para utilizar

Las soluciones de software y hardware que se pueden utilizar en la implementación de una plataforma IoT son amplias, debido a su interés para aplicar esta tecnología a nivel de la industria y empresa. En la actualidad existen varias empresas y fabricantes que se dedican a desarrollar sobre esta tecnología, para crear una infinidad de dispositivos para IoT que trabajan bajo plataformas que son de tipo propietarias. La tecnología de IoT se puede usar en diferentes campos de la industria para monitorear, controlar, interactuar entre cliente y máquina, comunicación entre máquinas (M2M) y toma de decisiones; lo que implica la optimización de recursos dentro de una empresa y mejoramiento de la productividad.

Para la implementación de la plataforma de IoT sobre la que se va a trabajar en el proyecto planteado, se puede ver de primera mano la solución hardware; que se puede utilizar para implementar en el diseño.

2.3.1 Tarjetas para IoT

El hardware utilizado para interconectar dispositivos hacia el Internet y tener un control se pueden encontrar fácilmente, algunos contienen software propietario para su configuración y otros son de software libre que permiten ser configurados fácilmente mediante lenguajes de programación de fácil aprendizaje.

En el mercado existen variedad de tarjetas que pueden ser utilizadas como, por ejemplo:

2.3.1.1 *Placa SmartEverything*

Esta placa se muestra en la figura 29, es una placa de desarrollo de IoT, se diferencia por tener un microcontrolador MCU de Atmel D21 de consumo ultra bajo, basado en el procesador de 32 bits ARM Cortex-M0+ y un módulo de comunicaciones Sigfox para la conectividad de IoT.

Figura 29. Placa SmartEverything, *(Interempresas.net, 2019)*.

2.3.1.2 *UrsaLeo Pi*

En la figura 29 se muestra una placa de desarrollo de IoT, se basa en el motor de computación Raspberry Pi, con el propósito de reducir costos y potenciar la mayor parte de capacidades del módulo de sensores Thunderboard 2 (Interempresas.net, 2019).

Figura 30. Placa UrsaLeo Pi, *(Interempresas.net, 2019)*.

2.3.1.3 *Tarjeta Compute Module 3*

Esta tarjeta se muestra en la figura 31, es una placa de desarrollo IoT pensado especialmente para el desarrollo de sistemas integrados en aplicaciones industriales, integra además una memoria flash eMMC de 4 GB y presenta la misma asignación de pines que su predecesor, el Compute Module 1 (CM1) tiene una potencia de procesamiento con cuatro núcleos que incorpora un procesador de aplicaciones BCM2837 Broadcom de 64 bits, procesador de cuatro núcleos ARM Cortex-A53 con una potencia de hasta 1,2GHz y 1GB de RAM LPDDR2 (Interempresas.net, 2019).

Figura 31. Placa Compute Module 3 (CM3), *(Interempresas.net, 2019)*.

2.3.1.4 *Placa M0 Arduino Pro*

La placa M0 Arduino Pro mostrada en la figura 32, también es una placa para IoT, está compuesta por el microcontrolador Atmel SAM D21 de baja potencia con el núcleo ARM

Cortex M0+ de 32 bits, ayuda con el aporte de un conjunto de características altamente flexibles y un mayor rendimiento (Interempresas.net, 2019).

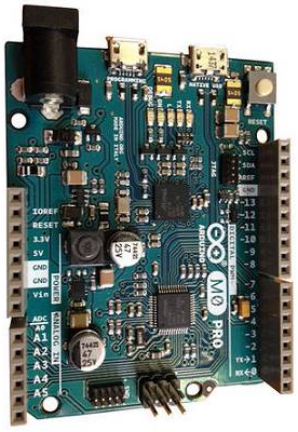

Figura 32. Placa M0 Arduino Pro, *(Interempresas.net, 2019)*.

2.3.1.5 *Placas de desarrollo*

Placas de desarrollo de microcontroladores mostrada en la figura 33, es una AVR de Microchip para Google Cloud. La solución de desarrollo permite la creación rápida de dispositivos seguros y conectados para aplicaciones de IoT. Esta solución de desarrollo rápido permite a los ingenieros conseguir el prototipado rápido de dispositivos conectados a la nube en cuestión de minutos (Interempresas.net, 2019).

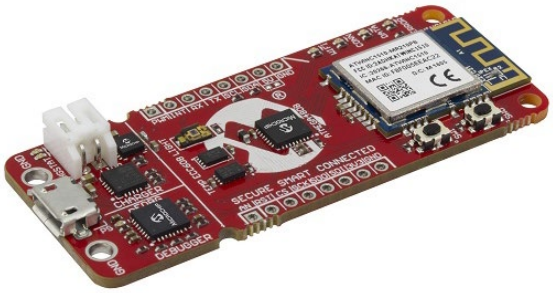

Figura 33. Placa AVR de Microchip, *(Interempresas.net, 2019)*.

Como se puede observar existe una variedad de tarjetas que se usan para implementar una plataforma IoT. Sin embargo, se estudiará dos tarjetas que permiten una implementación no muy compleja y sirven para realizar pruebas de concepto para poder entender su funcionamiento.

Se va a describir tarjetas conocidas y que actualmente son necesarias para realizar el prototipo, estas tarjetas son la de Raspberry Pi y la de Arduino. Estas tarjetas son sistemas

computacionales que fueron diseñados para propósito específicos con funciones dedicadas que pueden trabajar en tiempo real, para ponerlos en funcionamiento se deben programar y ajustar para cumplir una tarea específica que se requiere que hagan.

2.3.2 Placa de desarrollo Raspberry Pi

La placa Raspberry Pi es un dispositivo notable: es una computadora totalmente funcional de estructura electrónica pequeña y de bajo costo. Esta tarjeta es un ordenador de tamaño reducido que puede hacer cualquier cosa como una computadora grande pero no tan rápido. Puede ser usada para una variedad de proyectos y depende de los requerimientos para su configuración de software y puertos que se van a utilizar, para escoger el modelo de tarjeta que se debe adquirir. En la Figura 34 se puede observar la Placa de una Raspberry Pi 4, el modelo más actual hasta el momento, y como se puede ver cuenta con puertos USB 2.0 y USB 3.0, puerto Ethernet, mini HDMI, y otros módulos importantes como Wi-Fi y Bluetooth que se pueden usar para un proyecto específico.

Figura 34. Placa Raspberry Pi, *(Wikipedia, 2020)*.

De acuerdo a los antecedentes esta placa es de uso libre a nivel educativo como particular, permitiendo que cualquiera pueda distribuir esta placa, no se indica claramente si el hardware es libre, aunque en su página principal indican que disponen de contratos de distribución y venta con dos empresas (Wikipedia, 2020).

Actualmente se dispone de varios modelos de tarjetas que se pueden listar a continuación:

- Raspberry Pi 1
- Raspberry Pi 2
- Raspberry Pi Zero
- Raspberry Pi 3
- Raspberry Pi 3 Model B+
- Raspberry Pi 4 (Hardware actual)

Con el pasar de los años ha ido evolucionando y cambiando hasta el día de hoy, que se dispone de la tarjeta Raspberry Pi 4, para los diferentes proyectos que se realizan hoy en día se usa a partir del modelo 3, ya que su requerimiento procesamiento de software es alto, debido a que los requerimientos de las actualizaciones que se disponen hoy en día en la web son altos.

En la Tabla 5 se puede mostrar las características técnicas de cada uno de los modelos de tarjetas que existen actualmente y las que se pueden utilizar de acuerdo con los requerimientos que se tiene en un proyecto.

	RASPBERRY PI 3 MODEL B	**RASPBERRY PI 3 MODEL B+**	**RASPBERRY PI 4**
PROCESADOR	Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bit SoC	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC	ARM Cortex-A72
FRECUENCIA DE RELOJ	1,2 GHz	1,4 GHz	1,5 GHz
GPU	VideoCore IV 400 MHz		VideoCore VI (con soporte para OpenGL ES 3.x)
MEMORIA	1GB LPDDR2 SDRAM	1GB LPDDR2 SDRAM	1 GB / 2 GB / 4 GB LPDDR4 SDRAM
CONECTIVIDAD INALÁMBRICA	2.4GHz IEEE 802.11.b/g/n Bluetooth 4.1	2.4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 4.2, BLE	Bluetooth 5.0, Wi-Fi 802.11ac
CONECTIVIDAD DE RED	Fast Ethernet 10/100 Gbps	Gigabit Ethernet over USB 2.0 (300 Mbps de máximo teórico)	Gigabit Ethernet
PUERTOS	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación)	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación) Power-over-Ethernet (PoE)	GPIO 40 pines 2x micro HDMI 2x USB 2.0 2x USB 3.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Micro SD Conector de audio jack USB-C (alimentación)
FECHA DE LANZAMIENTO	29/2/2016	14/3/2018	25/7/2019

Tabla 5. Comparación tarjetas Raspberry Pi, *(Pastor, 2018)*.

En la figura 35 se muestra las partes principales de la Raspberry Pi, para poder tener una idea clara de los componentes físicos que tiene la tarjeta y de las funcionalidades que se tiene disponible para la implementación dentro de un proyecto.

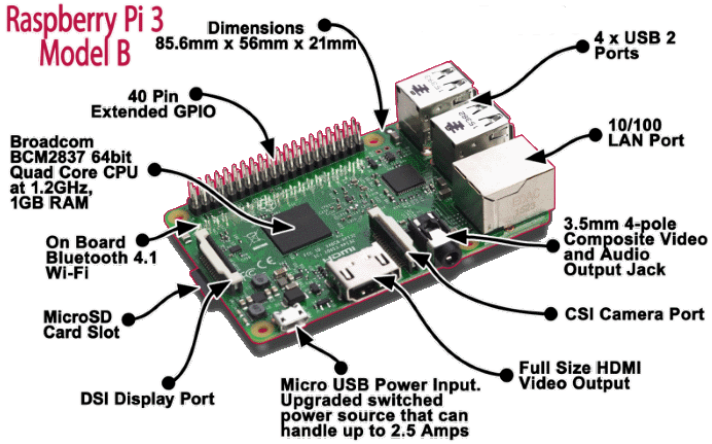

Figura 35. Partes físicas placa Raspberry Pi 3B, *(RaspberryShop, 2019)*.

En la figura 36 se muestra la distribución de pines que tiene de manera general para los diferentes modelos de tarjetas, mismos que se configuran en base al requerimiento del prototipo.

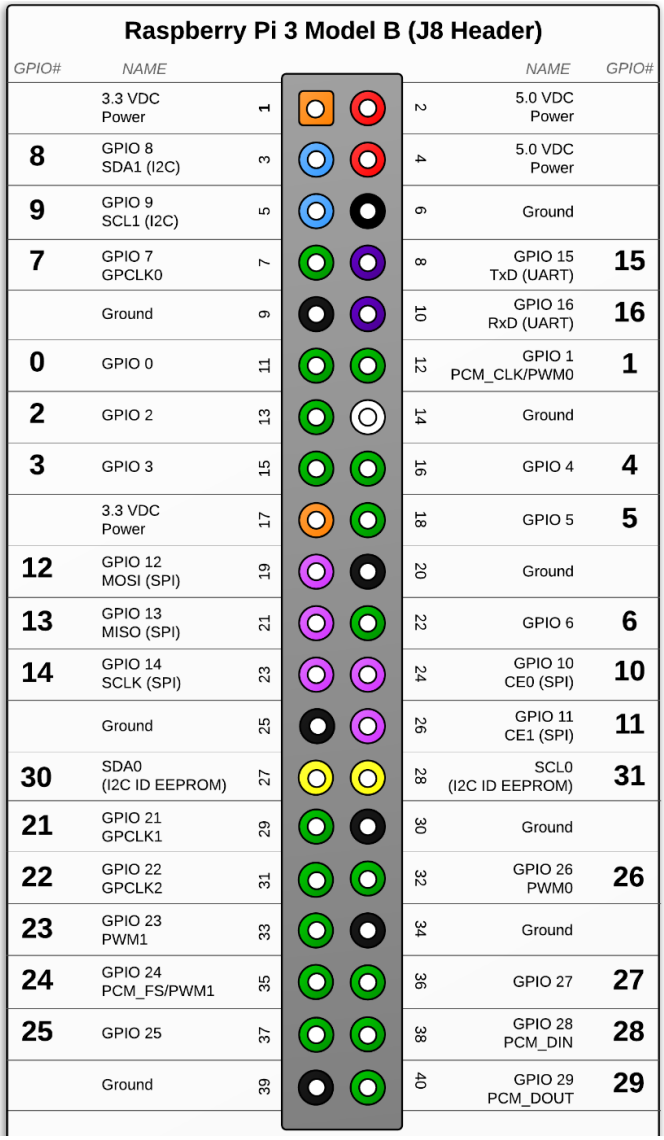

Figura 36. Distribución de pines de una placa Raspberry Pi Modelo 3B, *(Pi4J, 2019)*.

Una breve descripción de los pines se puede ver a continuación, los pines se configuran como entradas o salidas, donde se escriben valores digitales alto o bajo, pero se debe tener presente que el nivel alto es de 3.3V y no son tolerables a voltajes mayores de 5V.

Los pines 8 y 10 se configuran como interfaz UART para un puerto serie convencional, esta configuración viene por defecto dentro del sistema operativo Raspbian, debido a que UART se usa como consola (Moya Fernandez, 2017).

Los pines 3 y 5, se pueden configurar como interfaz I2C para interactuar con periféricos que siguen este protocolo. En el taller ya lo hemos configurado de este modo.

El pin 12 puede configurarse como salida PWM. En teoría los pines 12 y 13 pueden configurarse también como interfaz I2S (audio digital) pero hacen falta pines que no están disponibles fácilmente (Moya Fernandez, 2017).

Los pines 19, 21, 23, 24 y 26 se pueden configurar como la primera interfaz SPI (SPI0) para interactuar con periféricos que siguen este protocolo. En el taller ya los hemos configurado de este modo (Moya Fernandez, 2017).

Los pines 27 y 28 no están disponibles. Están reservados para la incorporación opcional de una memoria serie en las placas de expansión conforme a la especificación HAT. Son los únicos pines que en el arranque se configuran como salidas, todos los demás son configurados inicialmente como entradas para evitar problemas (Moya Fernandez, 2017).

Los pines 29, 31, 32, 33, 35, 36, 37, 38 y 40 proporcionan acceso a nuevas patas de GPIO que no estaban disponibles en los modelos originales. Estas patas pueden tener otros usos adicionales. Por ejemplo, los pines 32, 33 y 35 pueden utilizarse para salidas PWM (solo dos canales disponibles). Además, estas patas completan los pines necesarios para configurar otra interfaz SPI (SPI1), que no vamos a utilizar en el taller (Moya Fernandez, 2017).

2.3.2.1 *Usos de una Raspberry Pi*

Los usos de una Placa Raspberry Pi son varios, y se pueden describir los más destacados a continuación:

2.3.2.1.1. *Mini PC de escritorio*

Puede ser convertido en una mini PC de escritorio, ideal para realizar funciones de correo, navegación web, crear y editar documentos, sus puertos USB 2.0 permiten la lectura de dispositivos USB como un teclado y mouse, cuenta con un puerto HDMI para incorporar un monitor de video que permita visualizar y trabajar en el sistema operativo.

Para la conexión a Internet cuenta con un Puerto de red Ethernet y un módulo Wi-Fi que permite la conexión inalámbrica, además cuenta con un módulo Bluetooth 4.1 para la conexión con dispositivos que poseen este tipo de tecnología.

Con lo que respecta al sistema operativo, se debe trabajar con Raspbian, ya que este cuenta con herramientas de Ofimática, navegación y correo necesarios para un computador pequeño que ejecuta funciones básicas.

2.3.2.1.2. *Configura un servidor web o FTP con tu Raspberry Pi*

Esta es una aplicación útil que permite configurar la Placa Raspberry Pi como un servidor web, permitiendo alojar una página web propia. Una ventaja que se tiene es su bajo consumo de energía y espacio, se puede mantener encendido el servidor durante largo tiempo sin que esto afecte a la factura de luz que se paga mensualmente.

Para usar este tipo de aplicaciones un servidor se debe instalar los servicios correspondientes como: Apache, MySQL, PHP, PHPmyadmin o un paquete LAMP que integre todo en una gestión más sencilla, adicional se debe instalar archivos para un proyecto web si así lo requiere. También puede ser útil configurar dentro de la Raspberry Pi un servidor FTP para facilitar la transferencia de archivos (Andrés, 2018).

2.3.2.1.3. *Crea tu propio sistema NAS*

Esta tarjeta permite crear un propio servidor NAS para acceder a los archivos y datos desde la nube, para lo cual se requiere de un software de administración que debe ser instalado en la tarjeta y unidades de discos duros que se tenga disponibles. Un software compatible con Raspberry Pi es el NAS4free, este software pone al alcance de la Raspberry Pi todas las funcionalidades que se tiene en un Disco NAS comercial.

2.3.2.1.4. *Raspberry Pi como Media Center*

De igual manera que los anteriores usos, para sacar provecho de la placa Raspberry Pi se requiere la instalación de la plataforma de gestión de contenido Kodi, una versión compatible con la Raspberry Pi. Esto permitirá poner a disposición de la Placa todas las funcionalidades de un Media Center, para disfrutar todo el contenido multimedia como series y películas sin tener que invertir grandes sumas de dinero en equipos que cumplen con esta funcionalidad.

2.3.3 Placa Arduino

Arduino es una plataforma de prototipos electrónica de código abierto, el software y hardware son amigables de fácil uso. La placa de Arduino mostrada en la figura 37 consta de componentes electrónicos que se encuentran conectados a un microcontrolador para su funcionamiento, posee puertos que son específicos pensados para tipos de señales del tipo analógicos como digitales. Para la configuración requiere de un lenguaje de programación de bajo nivel. Es una herramienta de aprendizaje que permite la interacción con otras tarjetas como Raspberry Pi. Puede ser usada directamente como un interfaz para

conectarse directamente a una plataforma en la nube, permitiendo la transferencia de datos desde un sensor hacia un aplicativo Web donde se puede mostrar la información enviada a la nube.

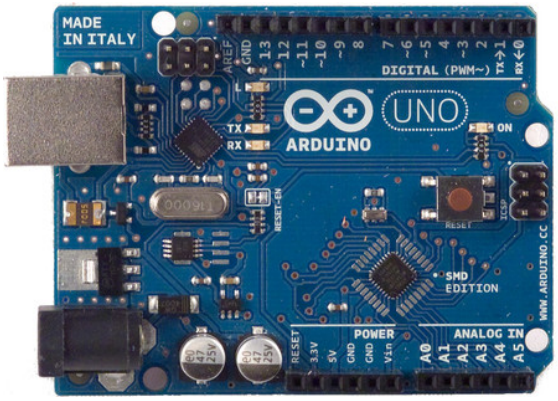

Figura 37. Placa Arduino Uno, *(Penalva, 2018)*.

Existen varias placas de Arduino que se diferencian por el modelo que se utilizara, algunas de las cuales se listan a continuación:

- Arduino Uno
- Arduino Mega
- Arduino Ethernet
- Arduino Due
- Arduino Leonardo
- Arduino Micro
- Arduino Mini
- Arduino Lilypad

Algunos de los modelos de placas descritos trabajan y son utilizados en base a las aplicaciones que se requieren; la diferencia de cada modelo está en sus componentes de Hardware que contiene cada una de las placas y el tipo de procesador que tienen incorporado para realizar las funcionalidades que se requiere. El más utilizado y completo para aplicaciones de IoT, es el módulo Arduino Mega que se muestra en la Figura 38.

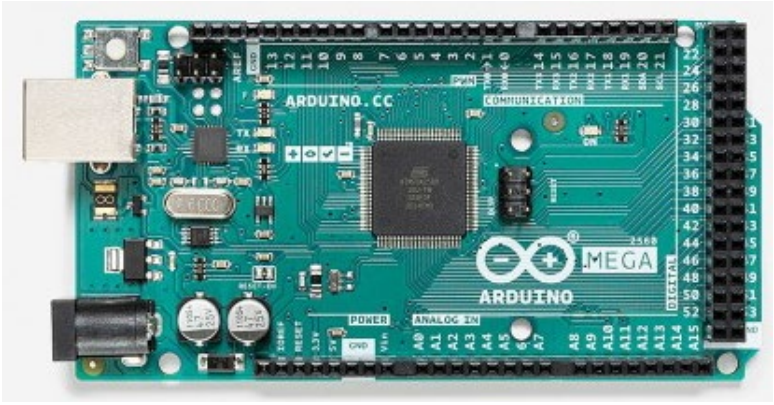

Figura 38. Arduino Mega, *(Arduino, 2020)*.

Arduino Mega 2560 es una placa de microcontrolador basada en el ATmega 2560, en comparación con Arduino Uno esta placa tiene 54 pines de entrada / salida digital, 15 de ellos se pueden usar como salidas PWM, 16 entradas analógicas, 4 UART que trabajan como puertos serie de hardware, un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, un encabezado ICSP, y un botón de reinicio (Arduino, 2020).

En la tabla 6 se listan los parámetros técnicos de la placa, para tener un óptimo funcionamiento del circuito que se implemente en un proyecto.

PARÁMETRO	DESCRIPCIÓN
Microcontrolador	ATmega2560
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límite)	6-20V
Pines de E / S digitales	54 (de los cuales 15 proporcionan salida PWM)
Pines de entrada analógica	dieciséis
Corriente CC por pin de E / S	20 mA
Corriente CC para Pin de 3.3V	50 mA
Memoria flash	256 KB de los cuales 8 KB utilizados por el gestor de arranque
SRAM	8 KB
EEPROM	4 KB
Velocidad de reloj	16 megaciclos
LED_BUILTIN	13
Longitud	101,52 mm
Anchura	53,3 mm
Peso	37 g

Tabla 6. Parámetros Técnicos de Placa Arduino Mega, *(Arduino, 2020)*.

En la figura 39 se muestra la distribución de pines de la placa Arduino Mega 2560, esta placa por su variedad de pines que posee se utiliza para circuito complejos y que requieren

gran capacidad de conexión de los sensores, actuadores e interruptores que permiten funcionalidades específicas.

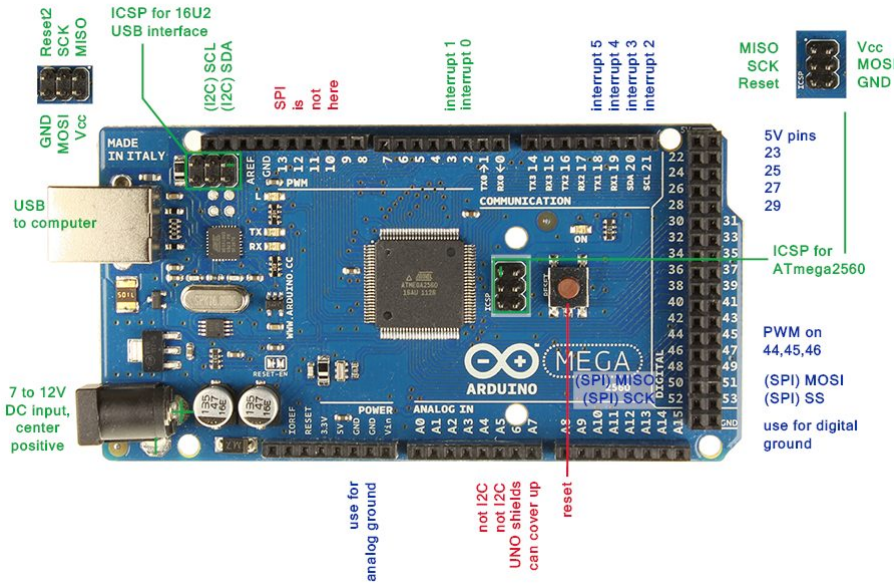

Figura 39. Pines Arduino Mega 2560, *(García Gonzáles, 2013)*.

2.3.4 Plataformas en la Nube para IoT

Existe una variedad de plataformas propietarias de hardware y software como servicio con las que se puede trabajar en la nube, algunas son de pago y otras son libres con funcionalidades limitadas. A continuación, se va a describir algunas de las más conocidas y que se pueden utilizar para implementar un sistema de IoT en las empresas.

2.3.4.1 *Cloud MQTT*

Es una plataforma en la nube, cuyo propósito es ofrecer la gestión y supervisión de un servidor de IoT, es una plataforma de pago con una solución para mensajes de IoT entre sensores de baja potencia y el cliente que puede ser una PC o un teléfono móvil como se puede observar en la figura 40.

Este servidor automatiza toda la configuración y ejecución de su agente mosquito alojado. Trabaja con MQTT, un protocolo de mensajería ligero ideal para la interconexión de sensores y aplicaciones móviles. Usa el cliente Cloud Websocket para ver los mensajes que se envía desde un dispositivo al navegador web y viceversa, cuya funcionalidad es ideal para mostrar los datos en tiempo real.

Figura 40. Cloud MQTT, *(CloudMQTT, 2020)*.

2.3.4.2 *Google Cloud*

Es una plataforma para IoT de pago, con una versión de prueba, que permite conectar dispositivos mediante un conjunto de herramientas con las que se puede procesar, almacenar y analizar los datos recogidos de los dispositivos, todo esto en la nube. Los servicios en la plataforma son escalables y administrados por el software integrado para tareas de procesamiento perimetral o local con capacidades de aprendizaje automático para satisfacer todas tus necesidades de IoT.

Figura 41. Google Cloud, *(Google, 2020)*.

2.3.4.3 *ThingSpeak*

Es una plataforma para IoT en la nube de pago, que permite el análisis de datos enviados por los dispositivos. Esta plataforma permite agregar, visualizar y analizar flujos desde dispositivos propios como las tarjetas de Raspberry Pi y Arduino que son las más conocidas por su configuración no compleja que mantienen, se puede tener datos en vivo y recibir alarmas o alertas que indican una variación en los datos recibidos desde los dispositivos.

El análisis de los datos en la nube se realiza mediante Matlab, una poderosa herramienta utilizada para análisis matemáticos complejos; y en esta ocasión para proyectos con IoT.

Figura 42. Plataforma ThingSpeak, *(ThinkSpeak, 2020)*.

CAPÍTULO III

3. DISEÑO Y CONSTRUCCIÓN DEL PROTOTIPO DE MONITOREO

3.1. Diseño y construcción de un sistema de sensores que permita la recolección de información de operación de las estaciones repetidoras de radio y televisión.

En este subtema se realizará una revisión de los problemas más comunes que se pueden presentar en una estación y los que se pueden dar una solución de manera remota, ya que algunos de los problemas como daños en equipos y reemplazo de módulos y componentes electrónicos se debe realizar mediante un mantenimiento correctivo, con visita personal en la estación.

Se presentará los circuitos que se implementaran para cada sensor y que tipo de datos se va a encontrar en cada uno, estos datos serán interpretados por la tarjeta Arduino y posteriormente se enviara a una base de datos para almacenar estos datos y poner a disposición de cualquier cliente que requiera este tipo de información.

3.1.1 Diseño de los sensores y actuadores

Para el diseño del sistema de sensores se procederá a identificar los parámetros básicos y acciones que se deben tomar en cuenta cuando existe una falla en una estación repetidora. Por la experiencia que se tiene, los sistemas de radio y televisión que operan en una estación presentan fallas comunes tanto en los equipos que operan, como en la caseta; para ello se va a mencionar los posibles daños que se pueden presentar y en base a ellos plantear el tipo de sensor o actuador.

3.1.1.1 *Problema: Variaciones de Temperatura*

El exceso de temperatura en una estación puede ser ocasionado por la mala operación de los transmisores que operan para cada servicio dentro de la estación, ya sea este de radio o televisión. La alta potencia que transmite hace que la etapa de amplificación genere altas temperaturas, y para hacer que el transmisor se mantenga refrigerado, se dispone de un sistema de enfriamiento que se compone de blowers (ventiladores de potencia moderada),

mismos que al operar hacen que el aire frio existente en la parte frontal del transmisor ingrese y circule a través de un disipador de grandes dimensiones para enfriarlo y el aire caliente generado salga por la parte posterior como se observa en la figura 43. El aire caliente expulsado hace que las condiciones de temperatura en la caseta se eleven hasta cierto punto donde se estabiliza.

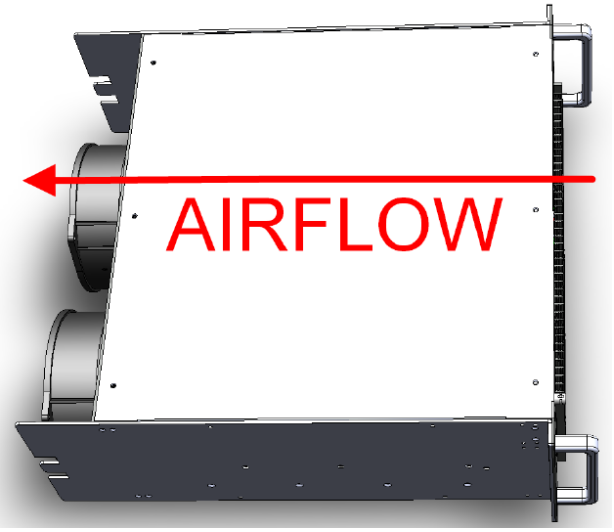

Figura 43. Circulación del flujo de aire en un amplificador, *(ScreenService, 2014)*.

El problema se ocasiona cuando la temperatura en la estación se eleva o bien baja; si se eleva el transmisor puede tener problemas con los blowers, que pueden dejar de operar por falta de mantenimiento; o bien la temperatura puede bajar debido a daños en los componentes eléctricos de los transmisores.

Solución: Sensor DHT22 para control de temperatura

Para controlar este tipo de variaciones de temperatura se realizará el diseño para un sensor que detecte la temperatura ambiente interne en la caseta y de acuerdo a la temperatura existente se trazará límites de máximos y mínimos, permitiendo mantener un nivel de temperatura acorde al funcionamiento de los equipos. Los datos serán transmitidos en tiempo real para mantener un estricto control de temperatura.

Para el diseño se utilizará el sensor DHT22 que se muestra en la figura 44, que es un sensor de temperatura y humedad de precisión con valores de medición muy cercano a la alta precisión, es fácil de implementar con cualquier microcontrolador y su señal de salida que entrega para el monitoreo es del tipo digital. Sus características técnicas se detallan en la tabla 7.

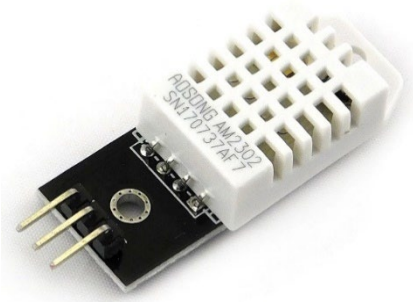

Figura 44. Sensor DHT22, (MercadoLibre, 2020).

SENSOR DHT22	
Alimentación	3.3VDC – 5VDC
Corriente máxima	2.5mA
Precisión Humedad	+/- 2%
Precisión Temperatura	+/- 0.5°C
Rango y unidades de Humedad	0% a 100%
Rango y Unidades de Temperatura	-40°C a 125°C
Velocidad de muestreo	0.5Hz (una muestra cada 2seg)
Dimensiones	15.1mm x 25mm x 7.7mm

Tabla 7. Características técnicas sensor DHT22 (GeekFactory, 2020)

Una vez conocido las características técnicas se procede con la interconexión del sensor, para este diseño se usará la tarjeta de Arduino Mega 2560, esta placa será quien almacene en uno de sus pines la información del sensor e interprete mediante un código escrito en lenguaje de programación, para obtener los datos de temperatura y posteriormente se enviará a una base de datos para que esté disponible para cualquier cliente que se subscriba a la plataforma.

La distribución de pines se muestra en la figura 45, se observa que cuenta de 3 pines que se conectan directamente a una fuente de voltaje y como salida se tiene el Pin 2 que corresponde a una salida de datos, que va directamente conectado al Pin 22 de la tarjeta Arduino Mega. El sensor es de bajo consumo de corriente, por este motivo la alimentación será suministrada por la misma placa de Arduino Mega para evitar utilizar una fuente de voltaje externa, la conexión del circuito se muestra en la imagen de la figura 46.

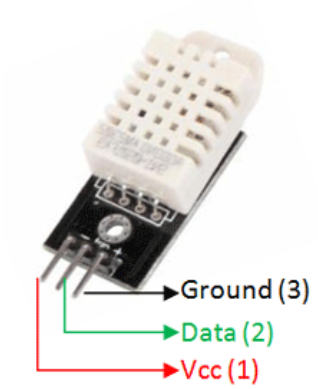

Figura 45. Pines Sensor DHT22 (Ja-Bots.com, 2020).

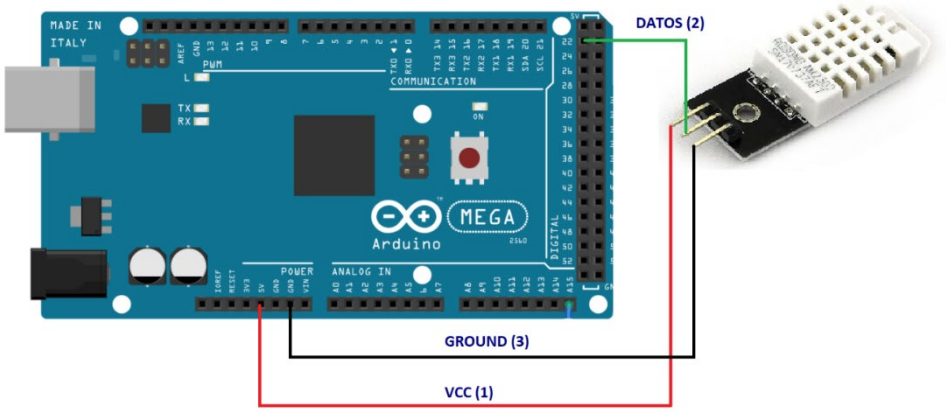

Figura 46. Conexión del sensor DHT22 (Elaboración propia).

Para la programación del código se utilizará el programa ARDUINO IDE, que es fácil de obtener desde la página de Arduino misma, este programa permite realizar la programación, compilar y cargar el código en la placa mediante un bus serial USB con la versión 2.0 o mayor.

Para poder trabajar con la librería del sensor DTH se debe descargar con el mismo software desde la pestaña **Herramientas>Administrador de Bibliotecas,** una vez entrado al meno aparece una pantalla en la que se debe buscar **DHT sensor library** como se muestra en

la figura 47, luego se procede con la descarga y queda listo para poder trabajar con el programa.

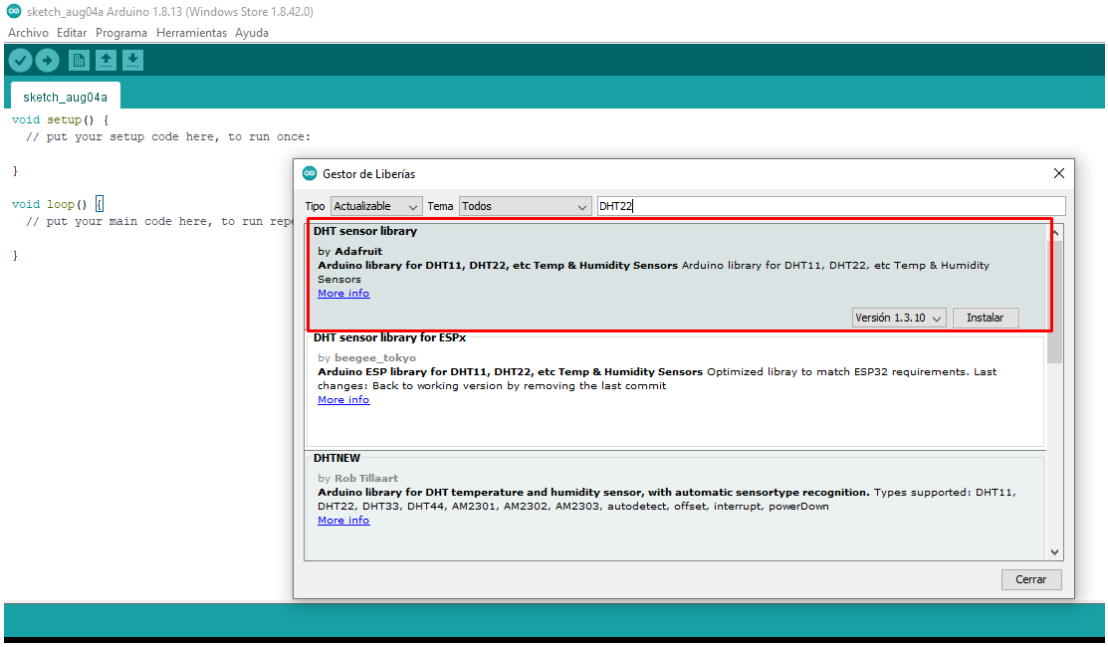

Figura 47. Descarga de DHT sensor library (O'Leary, 2020).

El código de Arduino para ejecutar el sensor de temperatura en la tarjeta se muestra en el ANEXO 1, detallado su programación utilizada para la obtención de datos que posteriormente se transmitirán.

Una vez listo el programa se sigue con la prueba correspondiente, para verificar que el programa funciona y recibe los datos del sensor se compila y carga el programa, para mostrar los datos obtenidos del sensor se utiliza el monitor serial que tiene el mismo programa de Arduino como una herramienta complementaria. El monitoreo de este sensor se puede verificar en la figura 48, donde se muestra el valor de Temperatura y Humedad obtenidas desde el sensor en tiempo real.

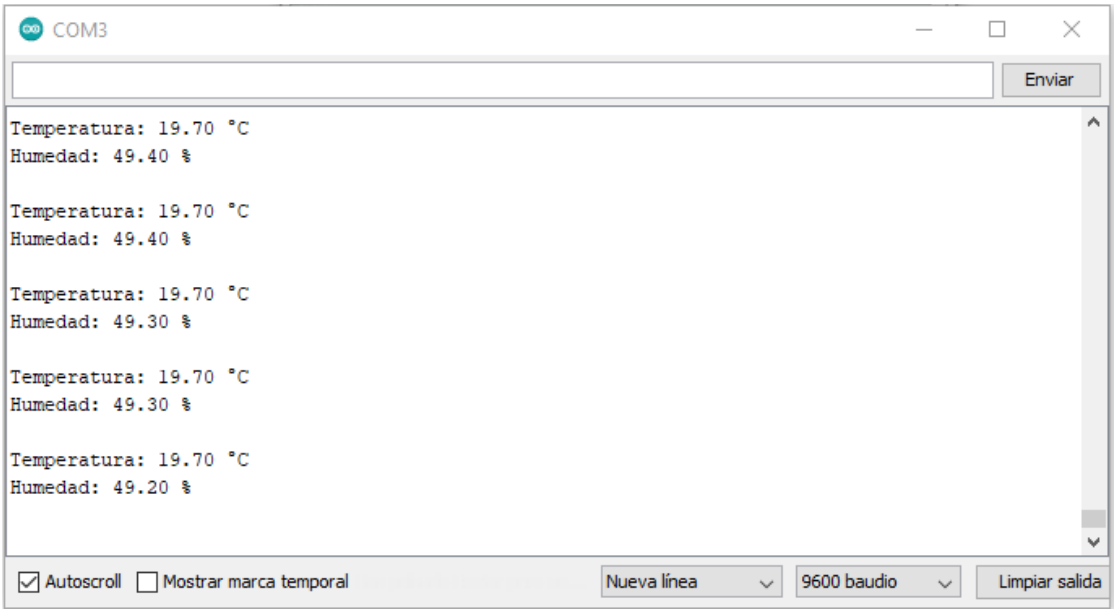

Figura 48.- Valores de temperatura y humedad obtenidos (Elaboración propia de captura).

Se puede observar el código de programa trabajando y ejecutando las funciones sin problemas, a este código se debe seguir incrementando funciones debido a que debe manejar más sensores para la recolección de datos y poder enviar hacia la tarjeta raspberry Pi quien será la encargada de publicar esta información al Internet y tenerla disponible desde cualquier punto.

3.1.1.2 *Problema: Daños en equipos*

Un problema adicional que se debe resolver es el daño en los transmisores, ya sea del servicio de radio o televisión. Una parte indispensable en el funcionamiento de los transmisores es conocer el nivel de potencia que transmite para determinar si este se encuentra operativo o dejo de funcionar por problemas en las etapas internas de los mismos.

Solución: Detectores de Radio Frecuencia en los transmisores

Para poder detectar la no operación de un transmisor por algún inconveniente, se deberá implementar un detector de Radio Frecuencia, que indique la potencia de operación del transmisor y poder determinar si existe problemas con los niveles de potencia a la salida. Para solucionar esta eventualidad el detector deberá permitir un cambio de operación a otro transmisor que se tenga de respaldo o caso contrario apagar el transmisor para evitar que este siga incrementando el daño interno de los componentes electrónicos.

Para el diseño se contará con dos detectores de Radio Frecuencia en el caso que la estación disponga de un transmisor de respaldo, caso contrario se trabajará con un solo detector de Radio Frecuencia que por lo general se da en sitios remotos.

Adicional el diseño debe permitir activar o desactivar los transmisores de forma remota para evitar daños en los componentes eléctricos hasta que se proceda con un mantenimiento correctivo por parte de la empresa encargada.

Todos estos datos generados por los sensores se guardarán en una base de datos para tener disponible esta información, a cualquier cliente que se suscriba a la plataforma de IoT, estos datos serán mensajes de alerta que indicaran que los transmisores dentro de una estación están dañados o dejaron de operar por algún inconveniente.

Un diagrama de bloques del circuito que se va implementar se muestra en la figura 49, donde se muestra dos detectores de Radio Frecuencia conectados a los transmisores (Principal y Respaldo), y los datos procesados en la tarjeta Arduino. En sus salidas se muestran controles que permitirán la activación y desactivación del transmisor de forma remota.

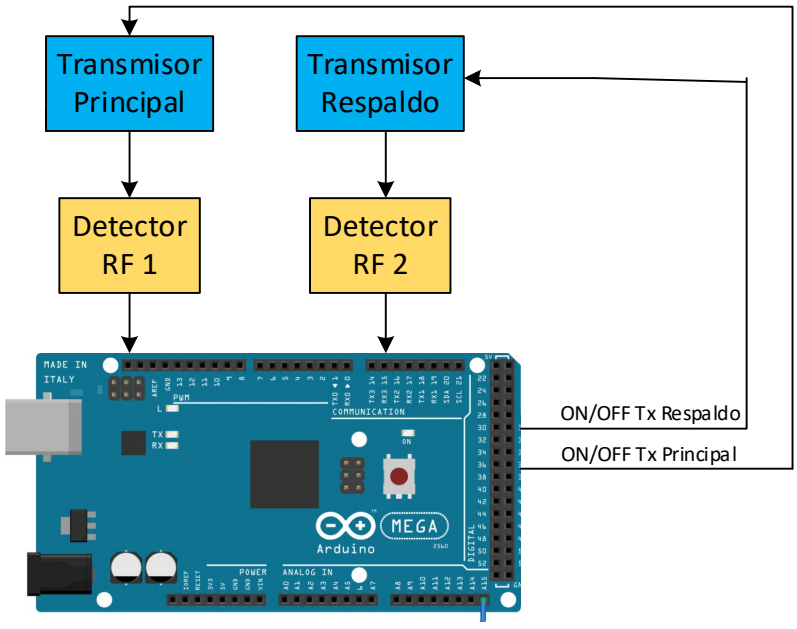

Figura 49. Sensores Detectores de RF para Transmisores (Elaboración propia).

El diagrama utilizara una tarjeta de 2 Relés, que, mediante la activación y desactivación, encenderán o apagaran la etapa de Radiofrecuencia en caso de tener algún problema alguna de las etapas del equipo.

El código mostrado en el ANEXO 1 ubicado al final del documento, en este se muestra la configuración de los pines e ingreso de los datos que se debe realizar para una conmutación automática de transmisores cuando en la estación se disponga de uno de respaldo.

3.1.1.3 *Problema: Cortes de energía de la Red Eléctrica*

Otro de los problemas que son importantes de analizar es cuando existe perdidas de fase en la red eléctrica que alimenta la estación en el cerro, existe ocasiones en las que por condiciones climáticas existentes no son favorables y existe cortes de energía en el cerro, por lo que no se puede conocer lo que sucedió. Es indispensable conocer cuando existe un corte de energía para poder ejecutar una acción correctiva inmediatamente y evitar que exista inconvenientes con el cliente.

Solución: Detector de fase eléctrica

Para la detección se va a implementar un supervisor de fase, conocido también como Monitor de Línea Monofásico que cumple la tarea de monitoreo de las fases y cuando exista ausencia de alguna desactiva un Relé, este Relé se usara para enviar un valor lógico de 1 hacia la tarjeta de Arduino Mega, quien interpretara esta información para enviar una alarma vía remota.

Para el diseño se contará con el Monitor de Línea Monofásico ICM491, que trabaja para la detección de fases y voltajes hasta de 220VAC entre fases, para el caso que se tenga una línea trifásica se podrá utilizar Detectores de Fase individuales que cumplirán la misma función de manera separada. Para el caso del diseño presentado se trabajará únicamente con dos fases, ya que la mayor parte de estaciones repetidoras que no son principales trabajan con Red Eléctrica de 220VAC.

El conexionado del Monitor de Línea se muestra en la figura 50, para un sistema eléctrico de 115VAC. Y puede trabajar con conexionado para 220VAC, voltaje utilizado en la mayor parte de estaciones.

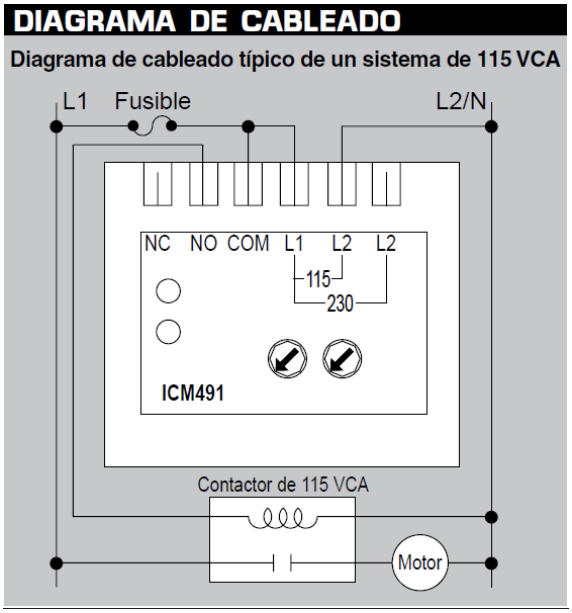

Figura 50. Diagrama de Cableado ICM491 (ICM-Controls, 2020).

Este detector se usará como un relé que permitirá enviar un valor lógico en alto cuando una de las fases tenga fallas, el diagrama de conexión del Monitor de fase se muestra en la figura 51, cuando una fase falla se activa un relé interno que permitirá enviar un voltaje DC hacia la Placa Arduino, para que este interprete como una falla en la Red eléctrica y se pueda determinar que no se tiene energía en la estación repetidora.

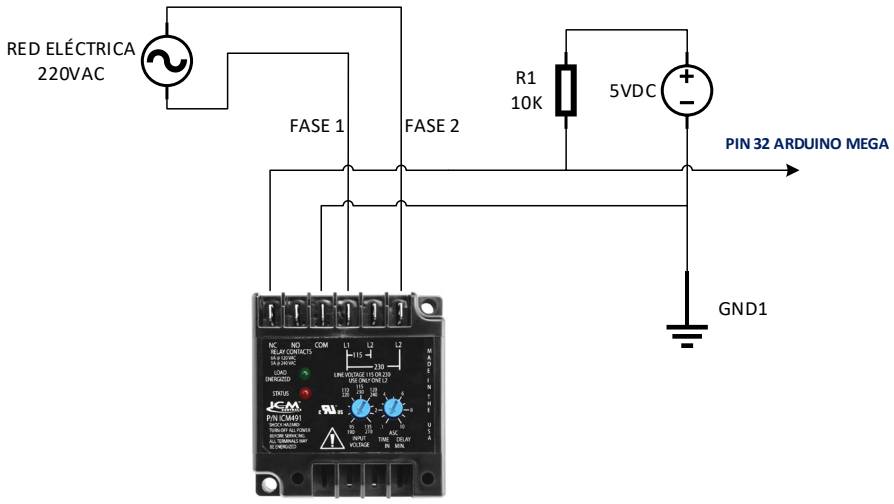

Figura 51. Monitor de fase con el circuito de aplicación (Elaboración Propia).

3.1.1.4 *Problema: Inhibición de equipos a causa de las altas potencia de Radio Frecuencia*

Un problema que se debe tener en cuenta en una estación repetidora que emite un alto nivel de potencia de RF, es cuando un equipo queda inhibido o congelado. Esto quiere decir que deja de estar operativo debido a que la Radio Frecuencia afecta la parte de procesamiento de señal que tiene este tipo de equipos, también pueden ser provocados por su largo tiempo de operación de 24 horas, 7dias a la semana y los 365 días del año. Dependiendo del fabricante y marca, unos son más sensibles que otros; sin embargo, para poder tenerlos operativo nuevamente se debe realizar un reinicio de los equipos afectados y todo vuelve a operar normalmente.

Solución: Reseteo de equipos mediante pulsos lógicos

Para solventar este problema se propone un sistema que permita de forma remota enviar un pulso para desactivar la parte eléctrica y volver a activar después de un determinado tiempo, sin que exista problemas en el reinicio de todo el sistema. Para ellos se propone un PIN de salida en la Placa Arduino que active y desactive un Relé como se muestra en la figura 52.

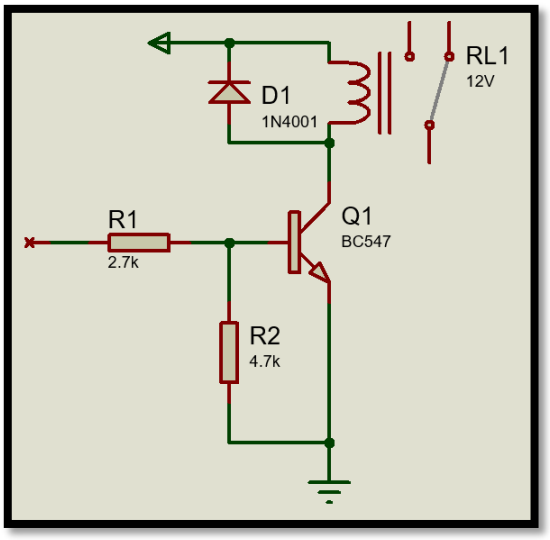

Figura 52. Circuito activar y desactivar Relé (Elaboración propia)

Como se puede observar en el circuito, la entrada por R1, es el terminal que va directamente hacia la salida de la señal de la Placa Arduino Mega, esta señal permitirá activar y desactivar un relé que estará conectado a un contactor principal del sistema de

distribución eléctrica para poder activar y desactivar el contactor, permitiendo cortar la energía eléctrica para que se produzca un reinicio de todo el sistema en la estación.

Una vez finalizado la revisión de los posibles problemas, los sensores y actuadores que pueden implementarse son de uso normal, que permiten realizar funciones básicas que son útiles al momento de resolver un problema y también monitorear parámetros para conocer el estado de manera general de la estación.

3.2. Análisis, instalación y configuración de un sistema IoT que permita el monitoreo y control en una estación repetidora de radio o televisión.

Una vez terminado el diseño de los sensores se procederá con la configuración de la plataforma de IoT para interconectar todos los sensores y almacenar información que esté disponible en todo momento, para la toma de decisiones como mantenimiento preventivos o correctivos que requiera la estación en algún momento.

3.2.1 Análisis

Para configurar la plataforma de IoT es necesario conocer que la comunicación se hará mediante el protocolo MQTT utilizado actualmente en este tipo de plataformas, debido a que es un protocolo ligero que optimiza el ancho de banda sin requerir un ancho de banda grande que impida el intercambio de información.

Se tiene dos opciones para el hardware de la plataforma:

La primera opción es configurar una tarjeta raspberry Pi, que se lo puede hacer trabajar como un servidor privado, el inconveniente es que se debe subir a la nube para poder acceder desde cualquier dispositivo externo. Para lo cual se debe ocupar un proveedor de DDNS, para poder identificar mediante un dominio al servidor.

La segunda es la instalación del bróker MQTT en un servidor en la nube que facilita la comunicación dependiendo de la configuración que se haga en el servidor. El servidor operará 24/7 ya que estará interactuando entre los dispositivos clientes de suscripción y clientes de publicación. Para esto se puede ocupar la plataforma de DigitalOcean, que es

un proveedor que alquila infraestructura en la nube, es de pago y su valor depende de las características del servidor creado.

Cualquiera de las dos opciones que se utilice, requiere tener instalado LAMP que es el acrónimo de los componentes necesario para instalar el servidor (Linux, Apache, MySQL, phpmyadmin) en el sistema operativo para gestionar la base de datos, servicio web en el caso de requerir crear una página web para mostrar la información, adicional se requiere instalar un servicio FTP para la transferencia de archivos en el caso de requerirlo. Una vez instalado los servicios complementarios se debe instalar el bróker MQTT para poder gestionar la transferencia de datos entre los diferentes clientes que se suscriban a la plataforma.

A continuación, se realizará la instalación de los servicios en las dos plataformas de hardware, pero más se enfocará a la segunda opción que es la que se va a utilizar en el presente proyecto, debido a circunstancias que se pueden presentar en una estación repetidora como un corte de energía eléctrica. En caso de presentarse esta situación, el servidor si se instalara dentro de la estación repetidora quedaría deshabilitado, por este motivo se utilizará un servidor en la nube. Únicamente se requerirá un pequeño UPS que permitirá la operación de los sensores por un lapso de 60 min aproximadamente, tiempo suficiente para indicar que hubo un corte de energía eléctrica y que se debe tomar acciones necesarias para levantar todo el sistema en la estación y ponerlo operativo.

3.2.2 Descripción de los Topics para la plataforma IoT

Se debe crear topics a los cuales se van a suscribir los clientes para ver la información publicada, así como también se crea topics para publicación de datos de los sensores. En la figura 53 se muestra los topics para publicación y suscripción que se van a configurar en los códigos de programación.

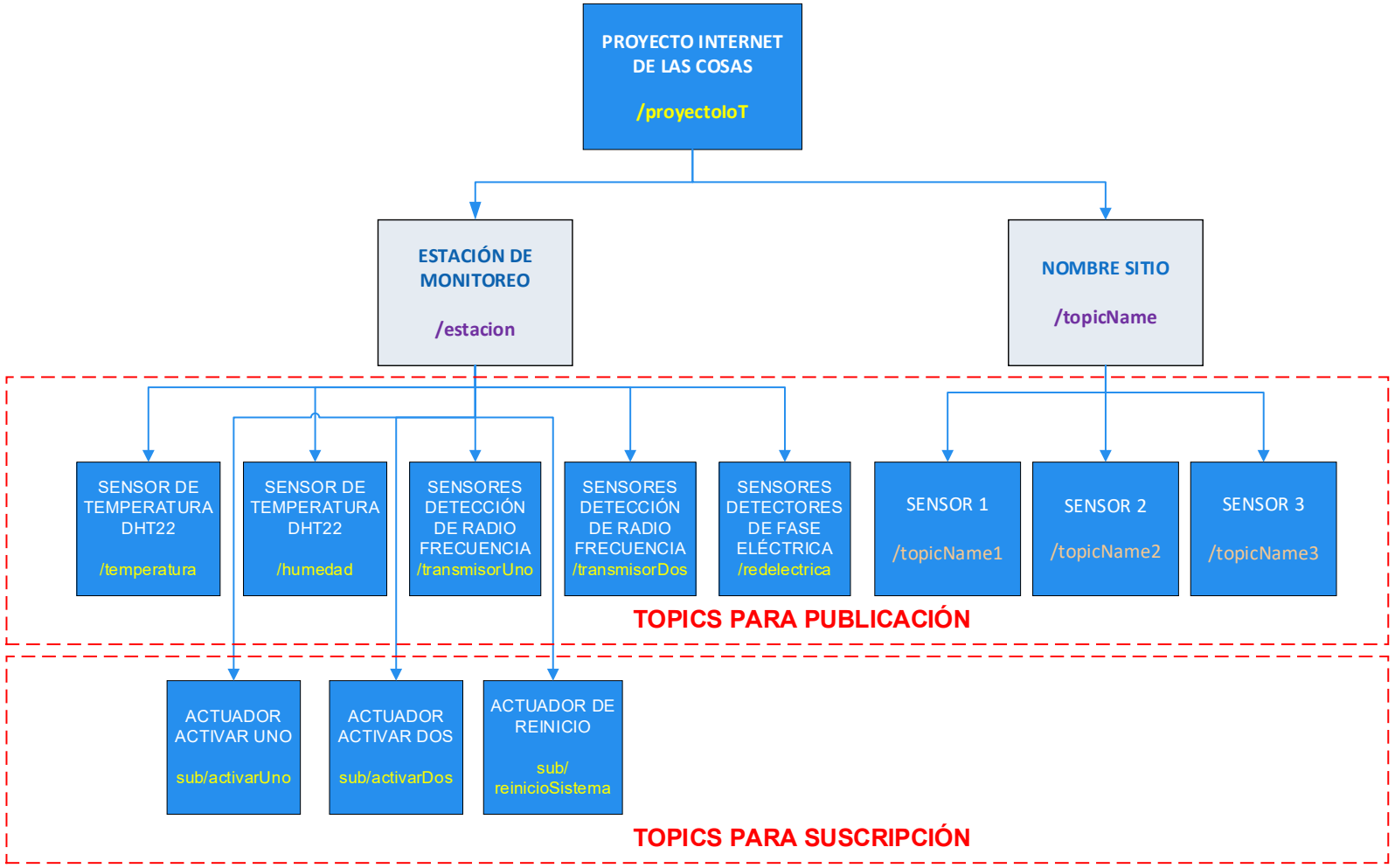

Figura 53. Distribución de Topics para configurar (Elaboración propia).

3.2.3 Instalación y configuración en Raspberry Pi

Se procede con la instalación de LAMP en la tarjeta Raspberry Pi, para tener un servidor con los servicios básicos que se pueden utilizar en el proyecto.

3.2.3.1 *Instalación de Software en Raspberry Pi*

La instalación del sistema operativo en la tarjeta Raspberry Pi se muestra detallado en el ANEXO 2, ya que pertenece a un procedimiento previo que se debe hacer antes de configurar la tarjeta e instalar los servicios necesarios para operar.

3.2.3.2 *Instalación de Servidor Apache*

Una vez instalado el sistema operativo basado en Linux, se procede con la instalación del servidor apache2, para la instalación se ejecutará una serie de comandos en el terminal de la Raspberry.

Se procede con la actualización del sistema de archivos de la Raspberry Pi con el comando ***sudo apt-get update*** se procede con la descarga del paquete de archivos para la actualización y con el comando **sudo *apt-get upgrade*** se procede con la lectura de los paquetes descargados para iniciar la actualización como se muestra en la figura 54. Para iniciar con la actualización se debe aceptar cuando indique el proceso. El proceso de actualización tardara unos minutos y una vez terminado como se indica en la figura 55 se procede a reiniciar la tarjeta Raspberry Pi para que se guarden los cambios.

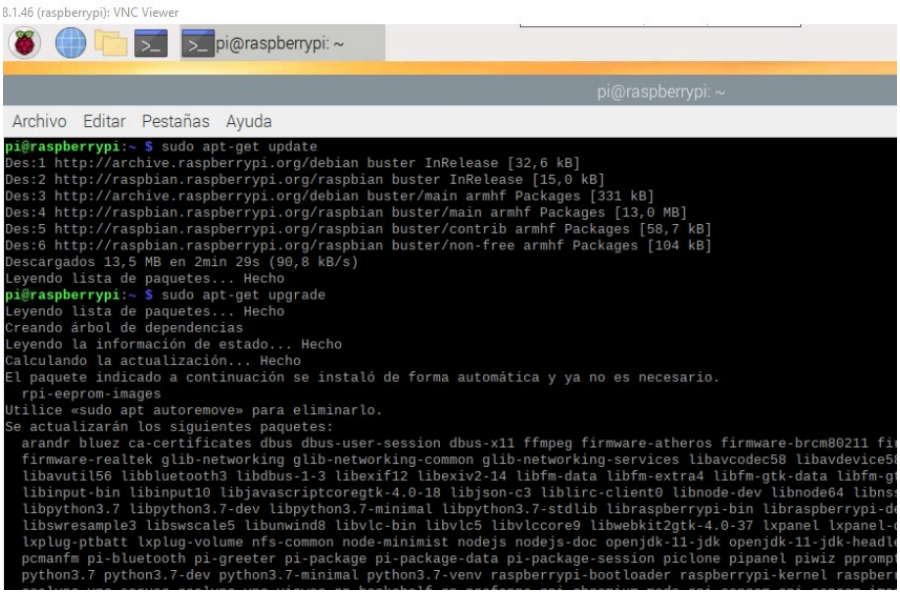

Figura 54. Comando para actualización del sistema (Elaboración propia de captura)

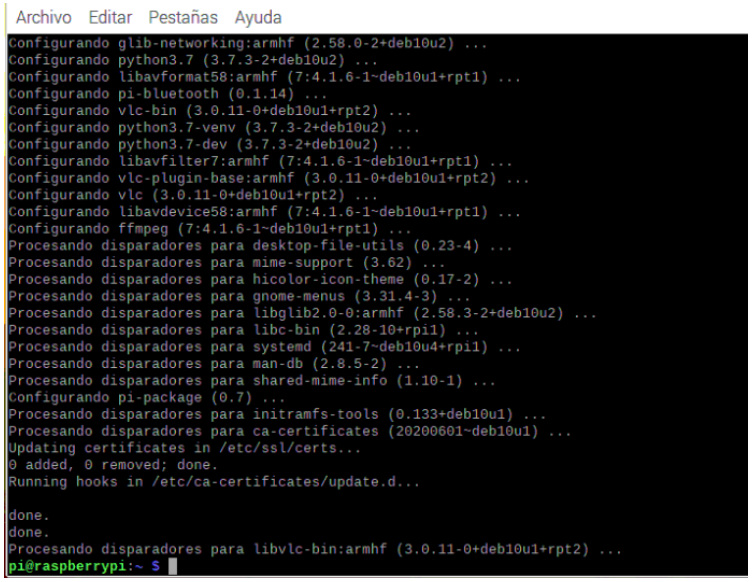

Figura 55. Actualización completa (Elaboración propia de captura).

Para la instalación del servidor apache2, se ejecuta el siguiente comando ***sudo apt-get install apache2***, una vez ejecutado el comando se debe aceptar la instalación y enseguida iniciara el proceso que tardara menos de un minuto, este proceso se muestra en la figura 56.

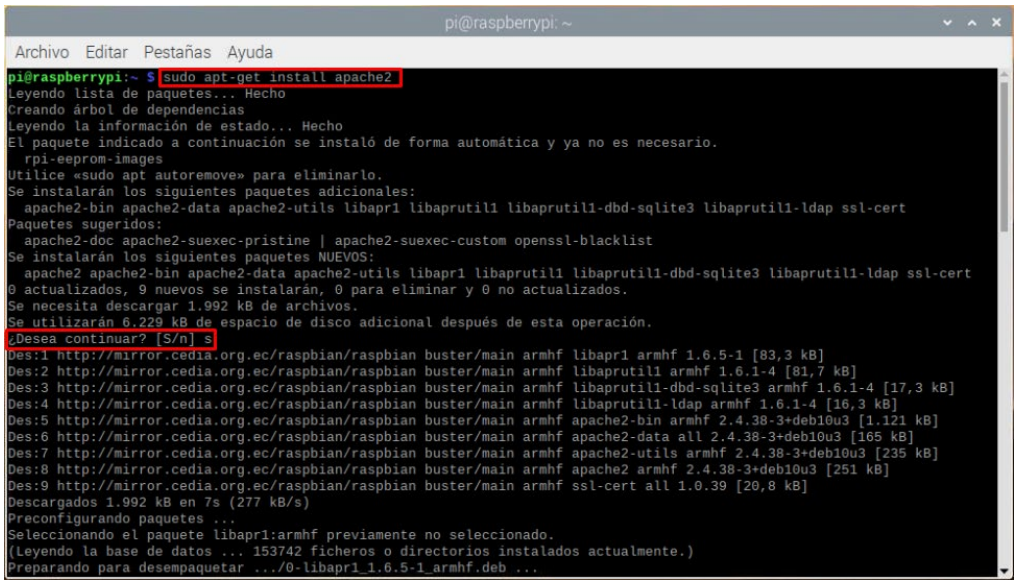

Figura 56. Instalación de Apache2 (Elaboración propia de captura).

Para verificar que el servidor está trabajando apropiadamente verificamos en la web con la IP de la tarjeta desde cualquier equipo cliente dentro de la red en la que se está trabajando y se puede observar la página de prueba por defecto mostrada en la figura 57.

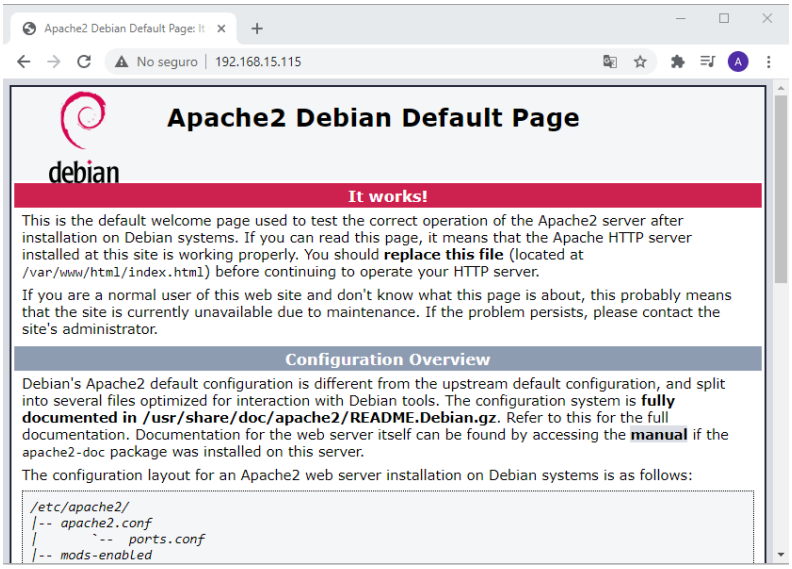

Figura 57. Página web de prueba por defecto (Elaboración propia de captura)

3.2.3.3 *Instalación de PHP*

Se sigue con la instalación del interprete PHP para el servidor apache, se ejecuta el comando sudo ***apt-get install php libapache2-mod-php***, de la misma manera se debe aceptar la instalación para que el proceso se ejecute como se muestra en la figura 58.

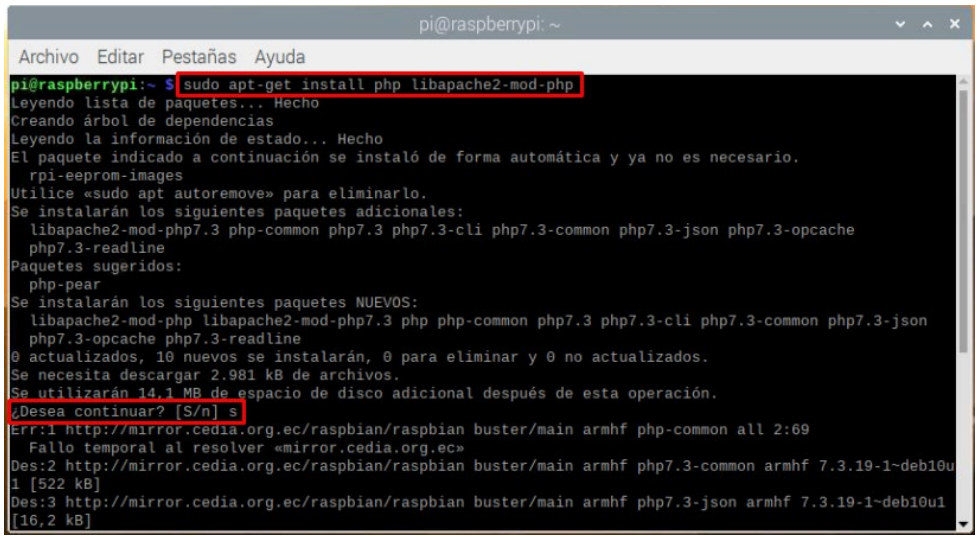

Figura 58. Instalación del interprete PHP (Elaboración propia de captura).

Para probar el funcionamiento de PHP se crea un archivo .php mediante el comando ***sudo nano /var/www/html/info.php***, en el archivo se escribe <?php phpinfo(); ?> y se guarda los cambios, para probar que está funcionando php se digita la dirección IP seguido del nombre del archivo creado, en la figura 59 se muestra el funcionamiento de PHP con la información misma de PHP.

Figura 59. Prueba de funcionamiento de PHP (Elaboración propia de captura).

3.2.3.4 *Instalación de base de datos MariaDB*

Luego de verificar la instalación del interprete php, se procede con la instalación del servidor de base de datos **mariadb**, para la instalación se ejecuta el comando ***sudo apt-get install mariadb-server***, se realiza la confirmación de la instalación como muestra la figura 60.

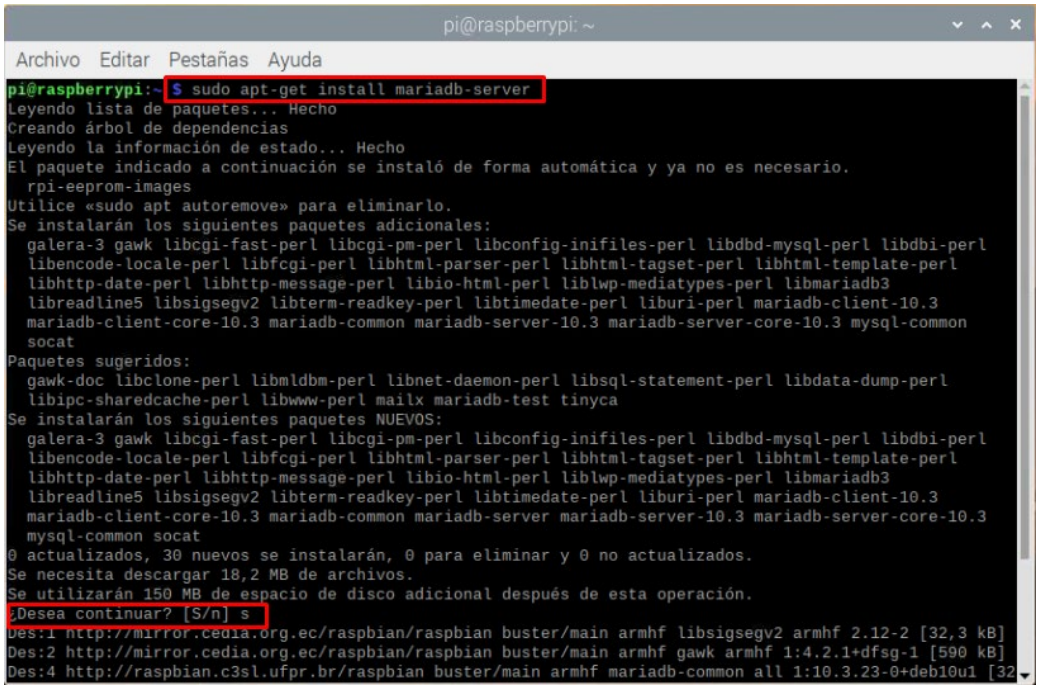

Figura 60. Instalación base de datos mariadb-server (Elaboración propia de captura).

Luego se cambia la configuración se seguridad que viene por defecto en el script mediante el comando ***sudo mysql_secure_installation***, ingresamos la clave del root para seguir con el cambio, se acepta el cambio de contraseña para el root y se ingresa una nueva, se elimina los usuarios anónimos y se acepta, se desactiva el acceso remoto al root,, se elimina la base de datos de prueba y accesos, y finalmente se carga los privilegios de acceso a las tablas; todo este proceso de cambio de seguridad se muestra en las figuras 61 y 62.

Para comprobar la instalación y ejecución del servidor de base de datos instalado primero se reinicia el servidor apache2, con el comando ***sudo service apache2 restart***, y luego se ejecuta el comando ***systemctl status mariadb.service***. Para verificar que el servicio está ejecutándose, en la figura 63 se muestra el estado del servicio ejecutándose correctamente.

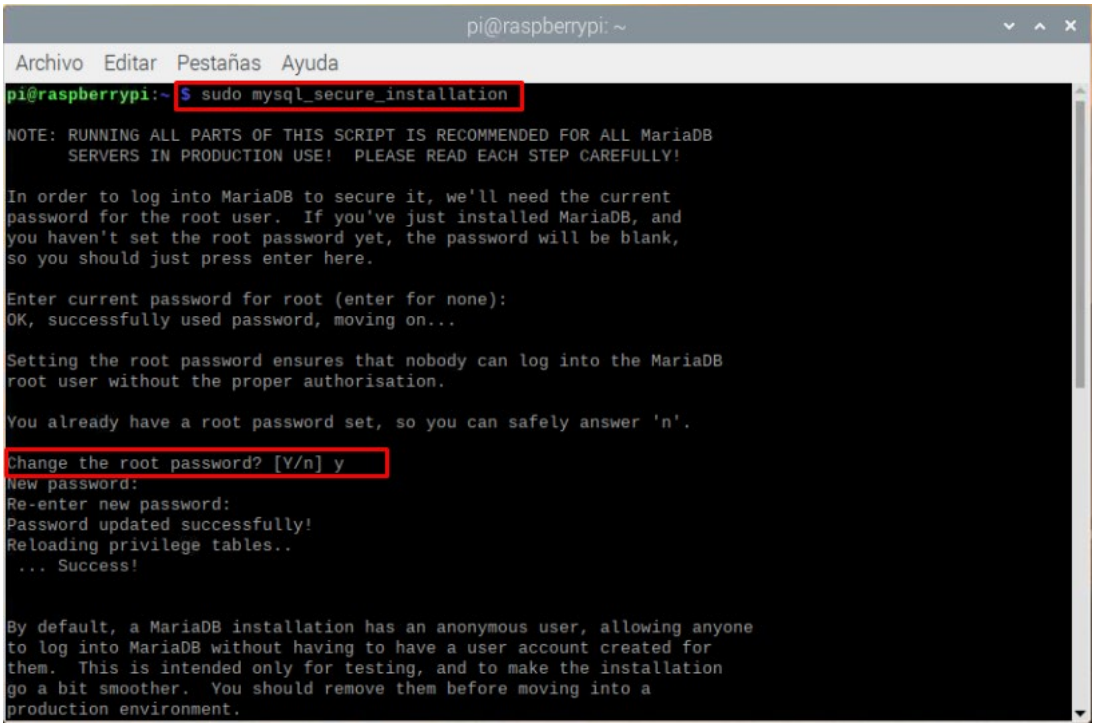

Figura 61. Cambio de seguridad de base de datos (Elaboración propia de captura).

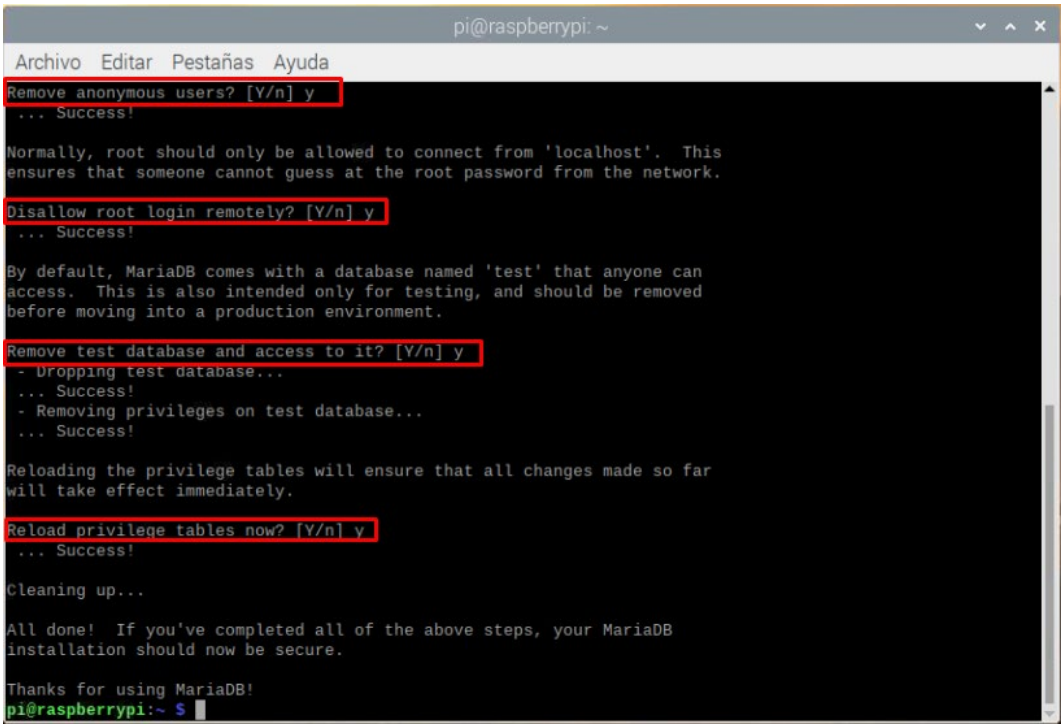

Figura 62. Cambio de seguridad de base de datos (Elaboración propia de captura).

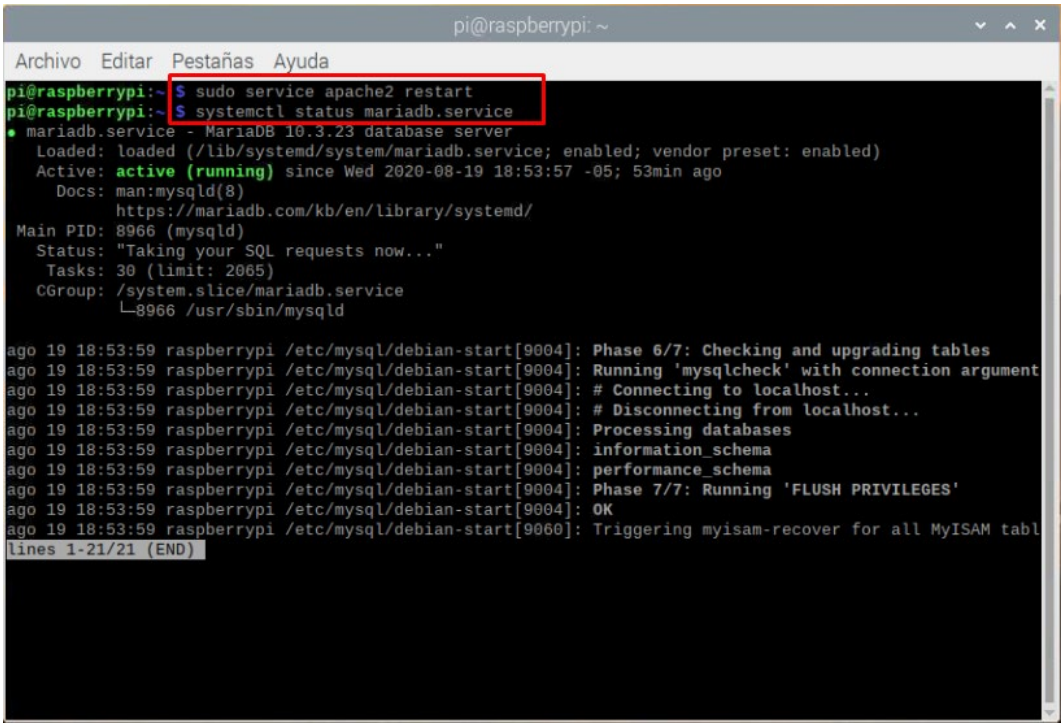

Figura 63. Ejecución del servicio base de datos mariadb (Elaboración propia de captura).

Para ingresar a la base de datos y configurar una tabla se ejecuta el comando **sudo *mysql -u root -P***, como se muestra en la figura 64, luego solicita la contraseña y se pone la que se configuró anteriormente en el cambio de seguridad, una vez adentro se procede a configurar.

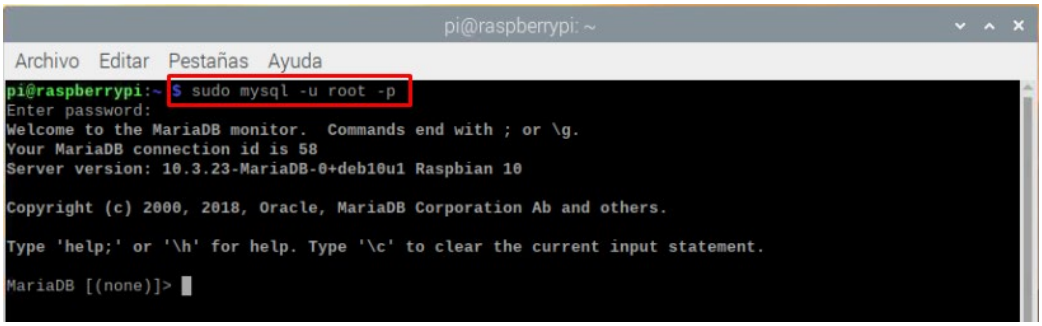

Figura 64. Ingreso a la base de datos mariadb (Elaboración propia de captura).

3.2.3.5 *Instalación de phpmyadmin*

Finalmente se instala la herramienta de software para manejar mariadb a través de la web, para la instalación se ejecuta el comando ***sudo apt-get install phpmyadmin php-mbstring php-gettext***, luego se confirma la instalación e inicia el proceso como se observa en la figura 65. Durante la instalación se solicitará una serie de configuraciones como

elección del servidor web mostrado en la figura 66, para este caso apache2; confirmación de configuración de la base de datos de forma automática mostrado en la figura 67; por último, se tiene la creación y confirmación de contraseña de phpmyadmin mostrados en la figura 68 y 69 respectivamente.

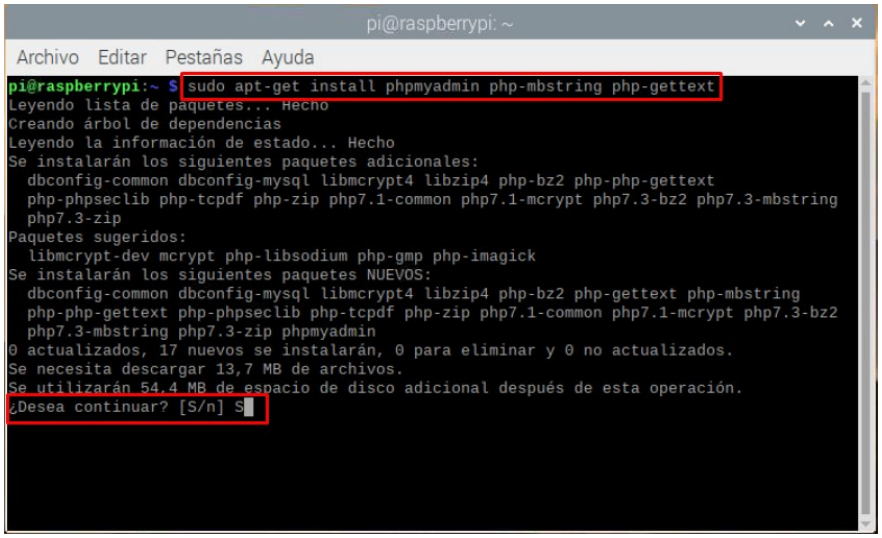

Figura 65. Ejecución comando para instalar phpmyadmin (Elaboración propia de captura).

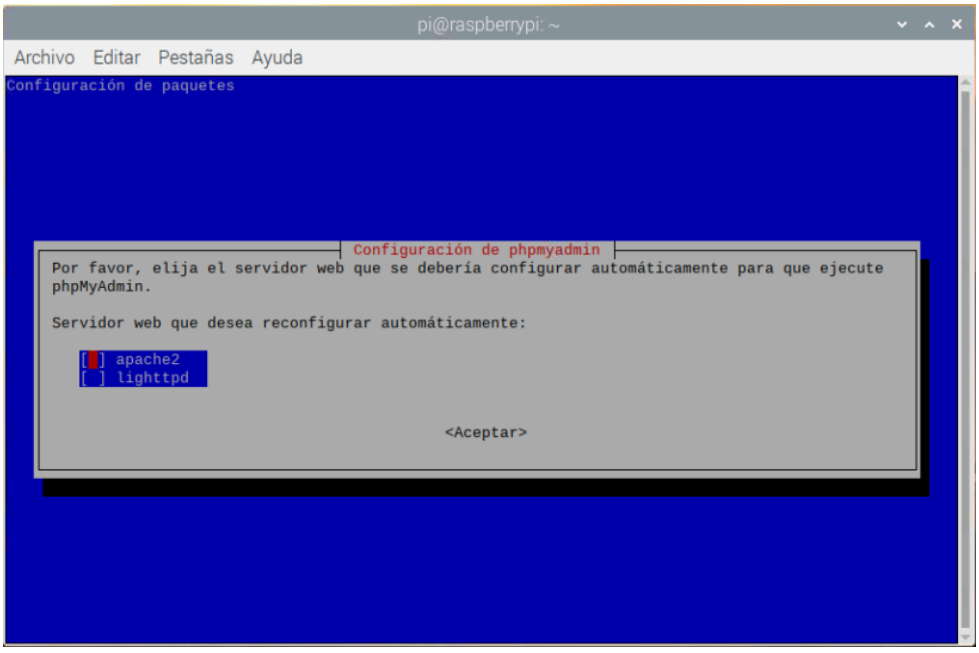

Figura 66. Elección del servidor apache2 (Elaboración propia de captura).

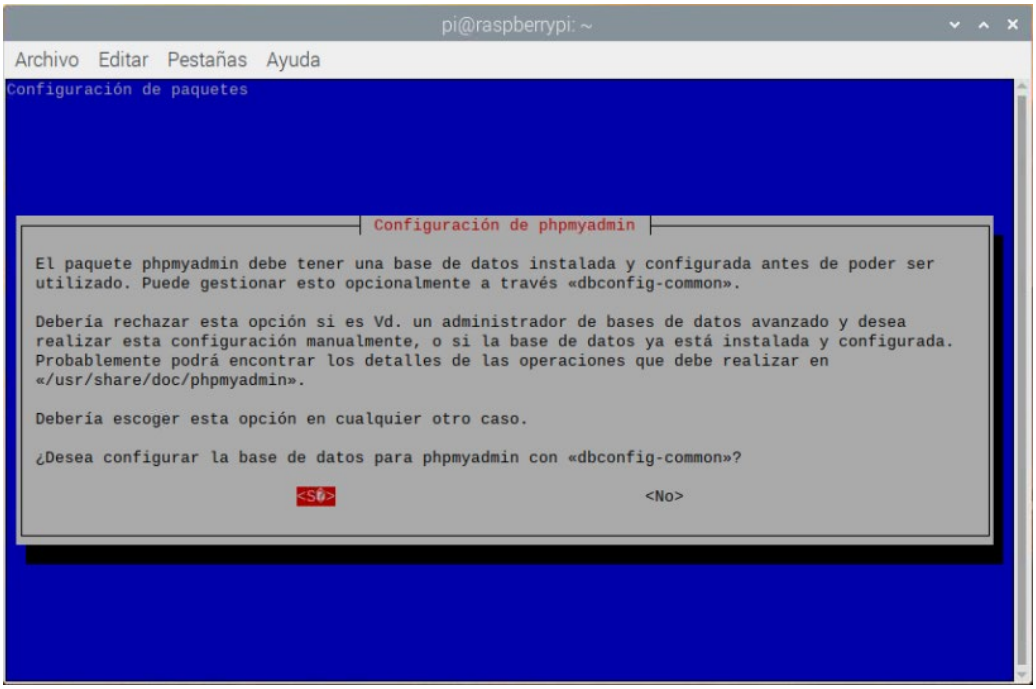

Figura 67. Confirmación de configuración automática de base de datos (Elaboración propia).

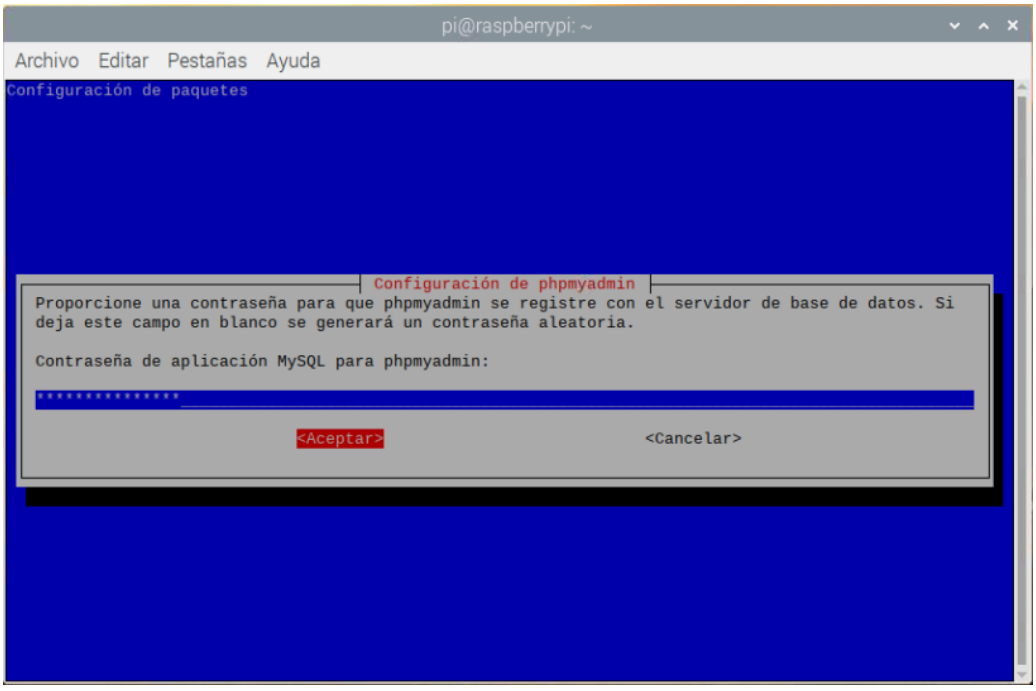

Figura 68. Creación contraseña para phpmyadmin (Elaboración propia de captura).

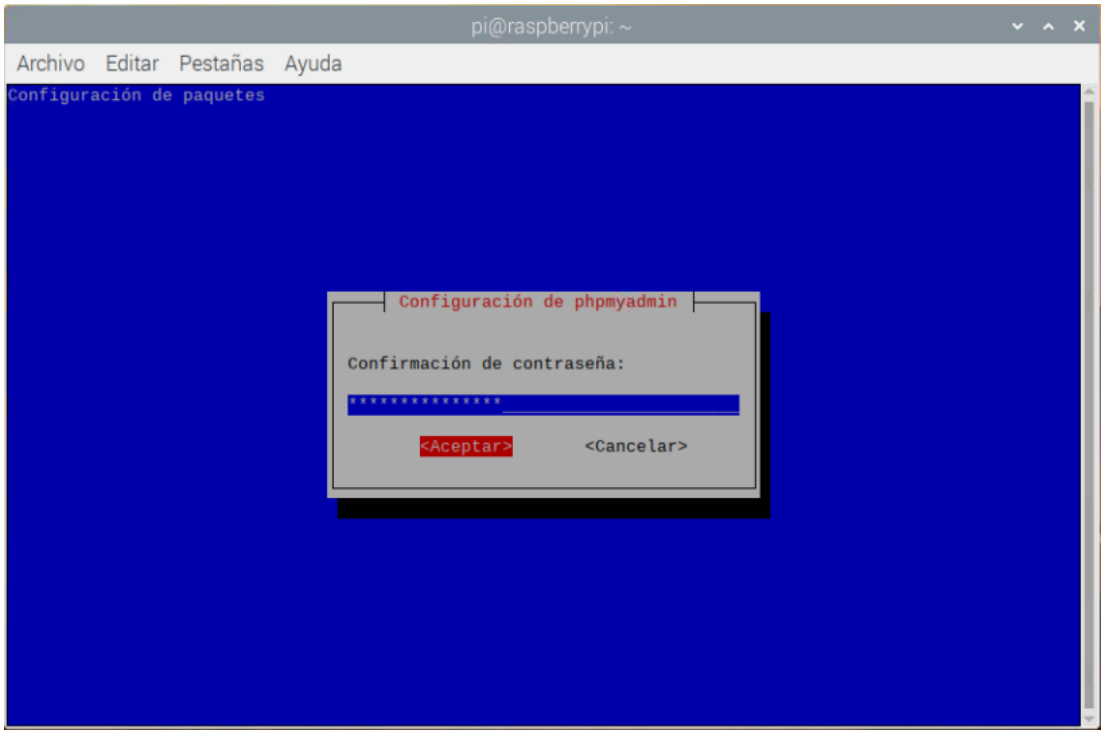

Figura 69. Confirmación de contraseña phpmyadmin (Elaboración propia de captura).

Hasta el momento no se ha creado el usuario para phpmyadmin, se procederá a crear el usuario ingresando a la base de datos con el comando ***sudo mysql -u root -p***, se crea el usuario y se da privilegios con las siguientes líneas de comando:

CREATE USER ‘raspberryiot’@’localhost’ IDENTIFIED BY ‘arduinoiot‘;

GRANT ALL PRIVILEGES ON *. * TO ‘raspberryiot’@’localhost’;

FLUSH PRIVILEGES;

La ejecución de estos comandos se muestra en la figura 70, de la misma manera se prueba el funcionamiento de phpmyadmin ingresando en la web la IP seguido de phpmyadmin y se ingresa con el usuario y contraseña creados anteriormente, el ingreso se muestra en las figuras 71 y 72.

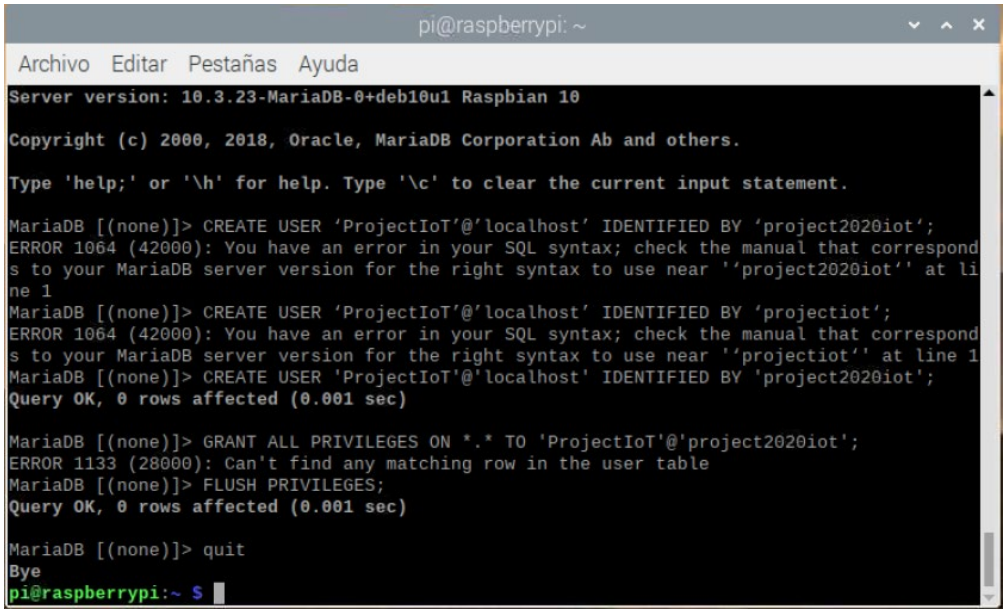

Figura 70. Creación usuario y contraseña para phpmyadmin (Elaboración propia de captura).

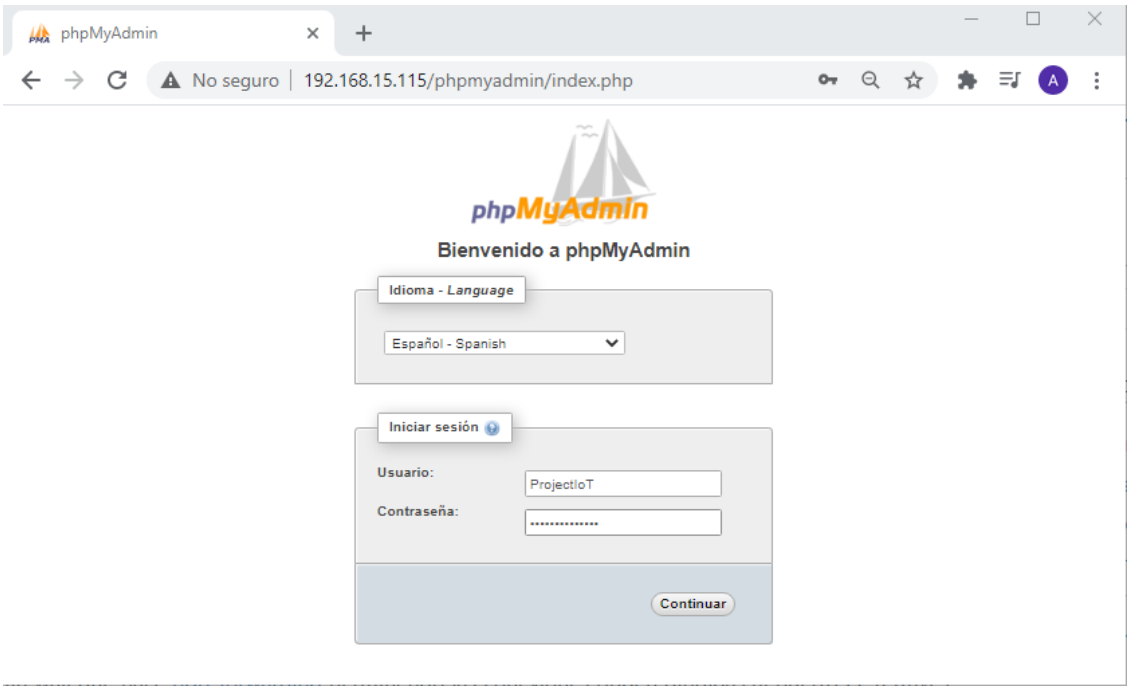

Figura 71. Prueba de ingreso a phpmyadmin (Elaboración propia de captura).

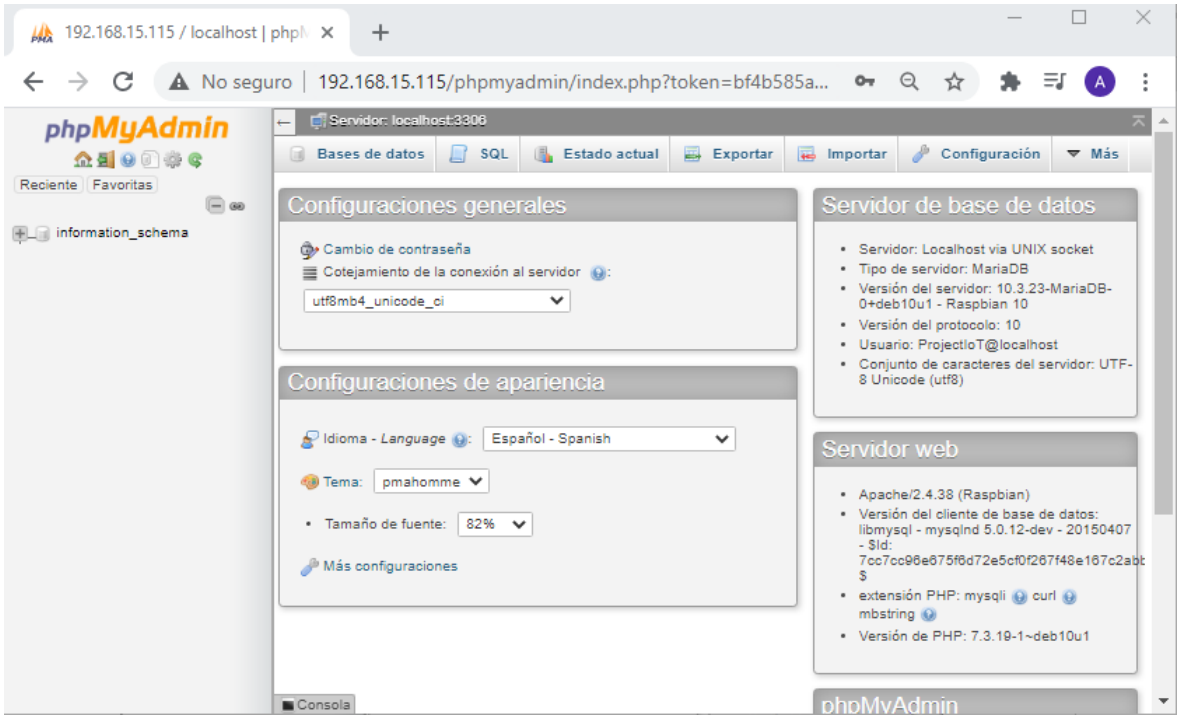

Figura 72. Página principal de phpmyadmin (Elaboración propia de captura).

Se realiza algunas configuraciones adicionales para phpmyadmin, en la dirección /etc/apache2 se edita el archivo apache2.conf con el comando **sudo nano /etc/apache2/apache2.conf** y al final del archivo se añade la línea Include /etc/phpmyadmin/apache.conf como se observa en la figura 73.

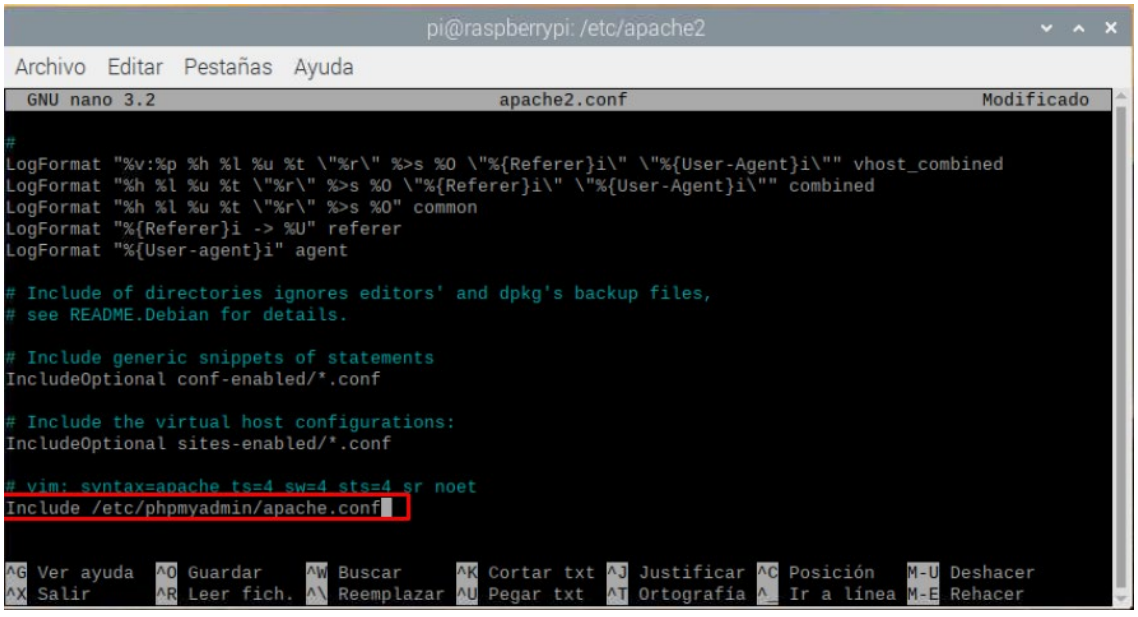

Figura 73. Edición del archivo apache2.conf (Elaboración propia de captura).

Se reinicia el servidor apache mediante el comando **sudo /etc/init.d/apache2 restart** como se ve en la figura 74.

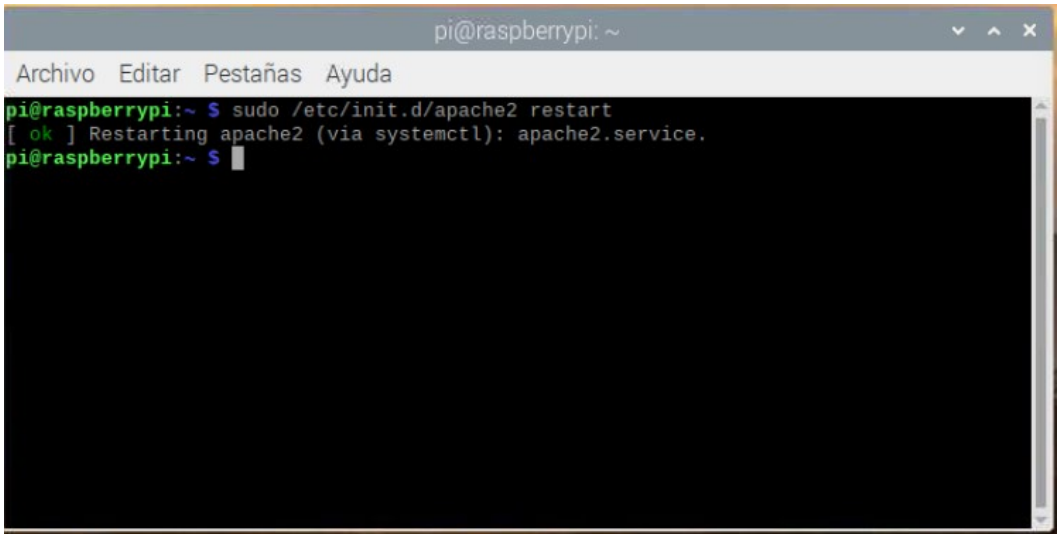

Figura 74. Reinicio del servidor apache2 (Elaboración propia de captura).

Luego, se edita el archivo php.ini en la dirección /etc/php/7.3/apache2/ mediante el comando ***sudo nano /etc/php/7.3/apache2/php.ini*** como se observa en la figura 75, en el archivo se añade la línea ***extensión=mysql.so*** después del campo “Dynamic Extensions” como se ve en la figura 76, para finalmente guardar los cambios y reiniciar la Raspberry Pi.

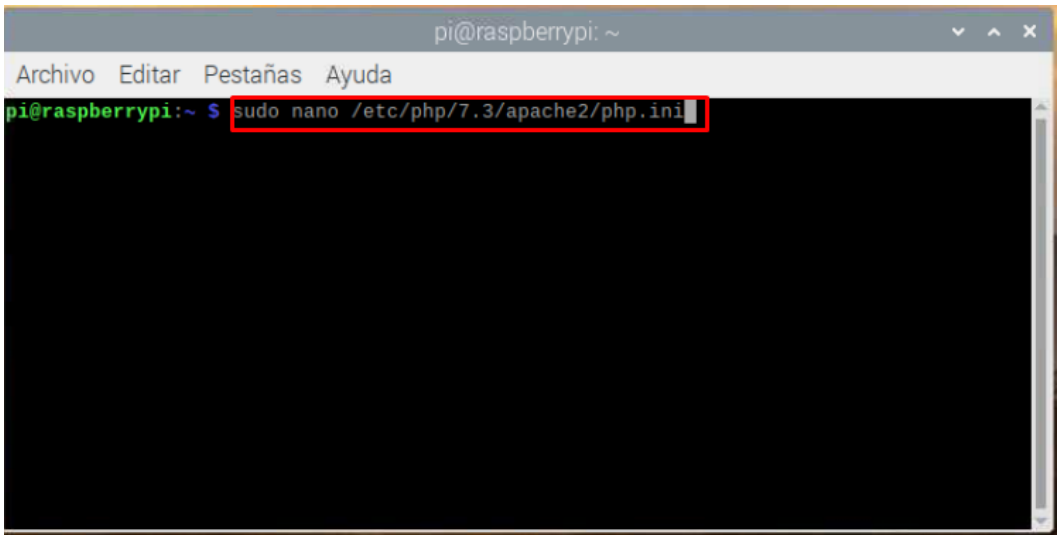

Figura 75. Comando para editar el archivo php.ini (Elaboración propia de captura).

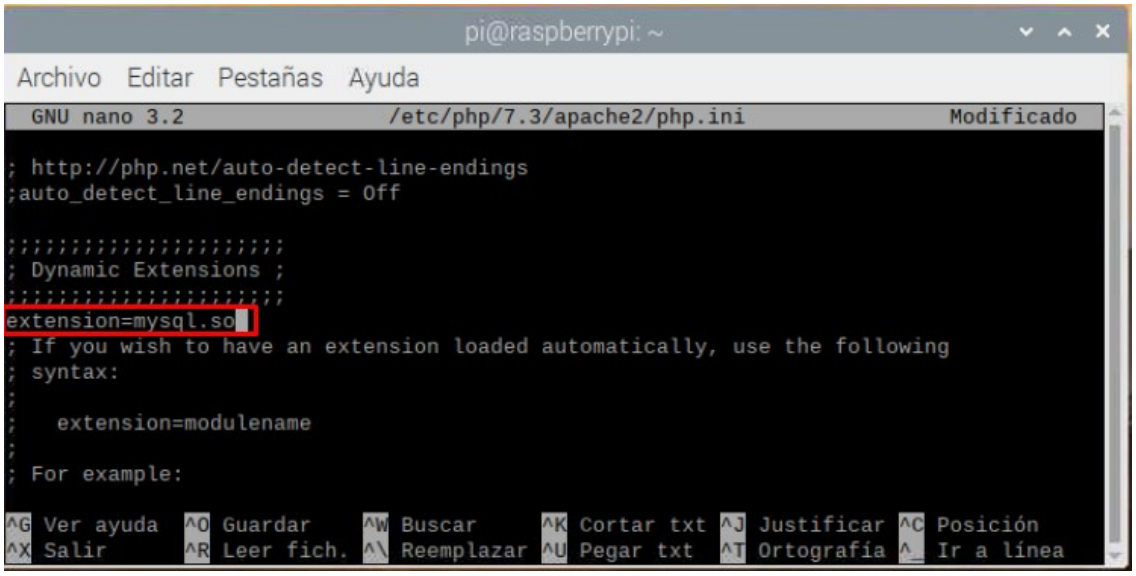

Figura 76. Añadir la línea adicional luego del campo Dynamic Extensions (Elaboración propia).

Se crea una carpeta de nombre conf.d en la dirección /etc/apache2/ mediante el comando ***sudo mkdir /etc/apache2/conf.d,*** una vez creado la carpeta se crea un acceso directo que también se conoce como enlace simbolico a esta carpeta.

Se ejecuta el comando ***sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf.d/phpmyadmin.conf*** para crear un enlace simbólico entre el origen de phpmyadmin y destino de apache2; luego se procede a actualizar el servidor con el comando ***sudo /etc/init.d/apache2 reload***.

Para el acceso al servidor apache se tiene como puerto predeterminado el 80, para brindar seguridad y distinguir de las aplicaciones que corren en la raspberry pi se procede con el cambio de puerto al 2401, para esto se edita los archivos en apache mediante los comandos ***sudo nano /etc/apache2/ports.conf*** y ***sudo nano /etc/apache2/sites-available/000-default.conf***, y finalmente de reinicia el servidor apache como se ve en la figura 77.

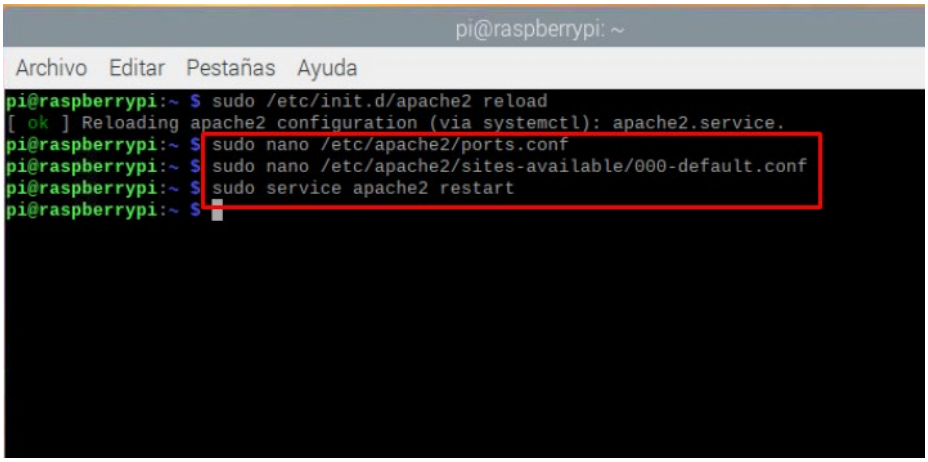

Figura 77. Cambio al puerto 2401 para el servidor apache (Elaboración propia de captura).

3.2.4 Instalación y configuración en Plataforma Digital Ocean

Digital Ocean es una plataforma de alquiler de servidores virtuales, que tiene su sede principal en New York, este proveedor alquila instalaciones de centros de cómputo existentes en sitios como New York, Toronto, Londres, San Francisco y otros más. Dentro de la plataforma se puede configurar un servidor completo que pueden montarse en la Nube y se puede acceder a los diferentes servicios directamente desde cualquier sitio, es una buena plataforma para realizar pruebas de conceptos como de protocolos, como el de MQTT que se utilizara en el presente proyecto.

A continuación, se presentará la creación del servidor y configuración de los servicios necesarios que se utilizaran para el presente proyecto.

3.2.4.1 *Creación del servidor en Digital Ocean*

Para comenzar, se debe crear una cuenta en Digital Ocean, el costo por utilización de la plataforma es de aproximadamente $5.00. Una vez creada la cuenta se crea un servidor en base a los requerimientos que se tiene, para empezar, se creara un servidor con el sistema operativo GNU/LINUX, la distribución que se utilizara será Debian 10 como se puede observar en la figura 78.

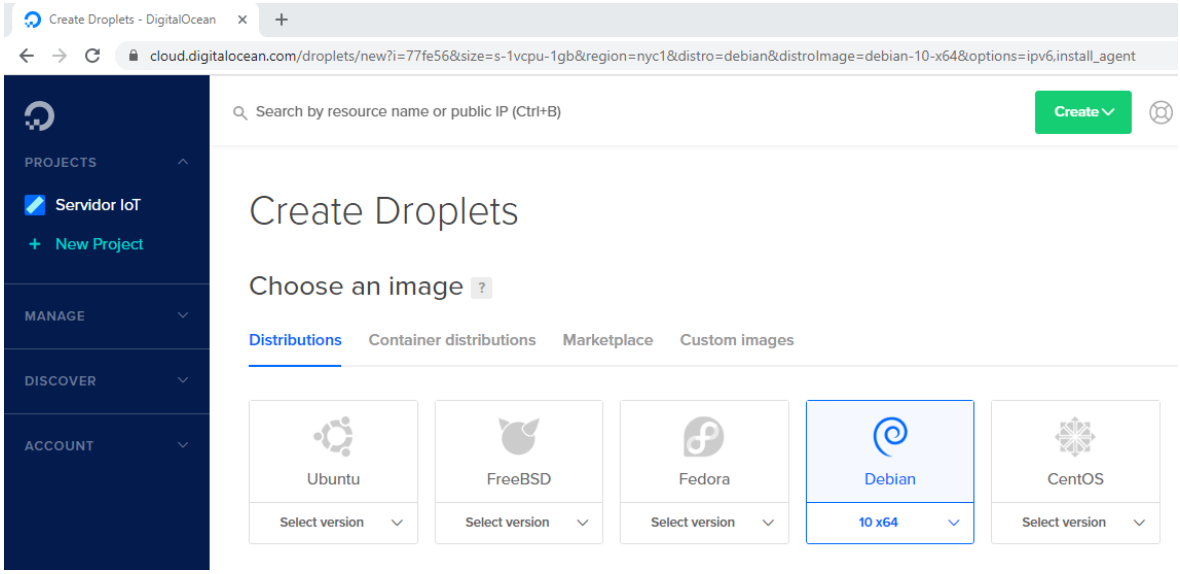

Figura 78. Selección de Sistema Operativo (Elaboración propia de captura)

Se escoge el plan básico, ya que limita sus funcionalidades y adicional el costo es bajo en comparación con los demás planes como se observa en la figura 79. Esta plataforma permite el acceso por SSH para la configuración mediante la línea de comandos. Luego se selecciona la ubicación del servidor físicamente, para ello se escoge una ubicación en New York como se puede ver en la figura 80.

Una vez terminado la selección de hardware y los costos debido a lo que se selecciona, se procede a configurar la clave de acceso por SSH, para la configuración se puede hacer de dos maneras, se puede generar un SSH key que es una manera segura o simplemente con un password.

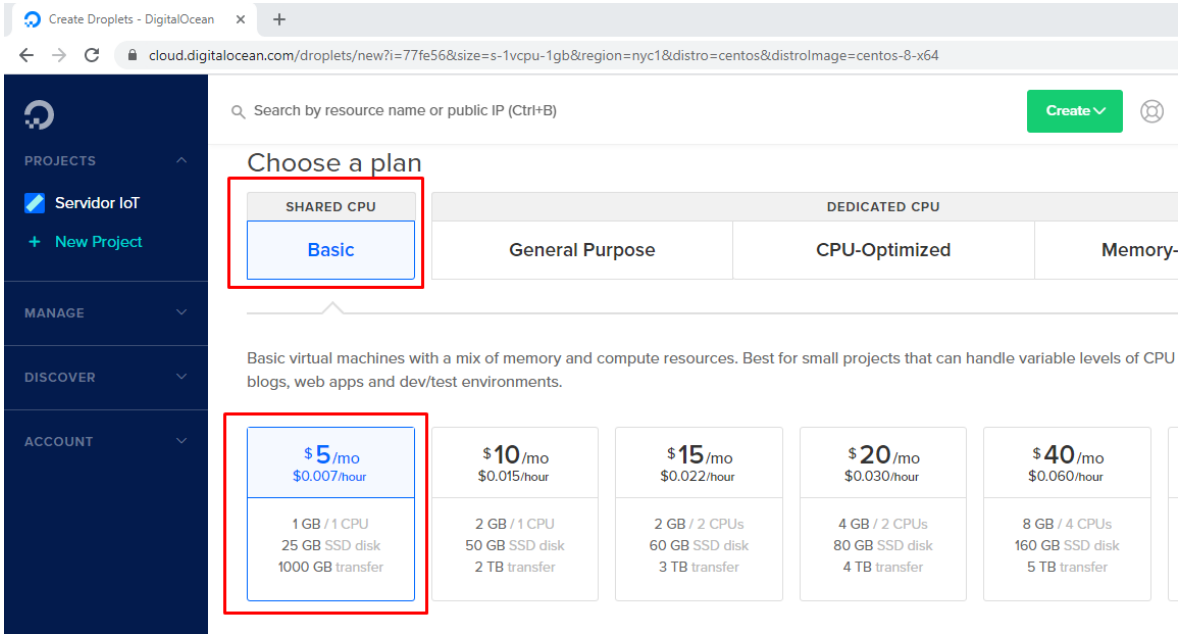

Figura 79. Selección del plan básico para utilizar el servidor (Elaboración propia de captura).

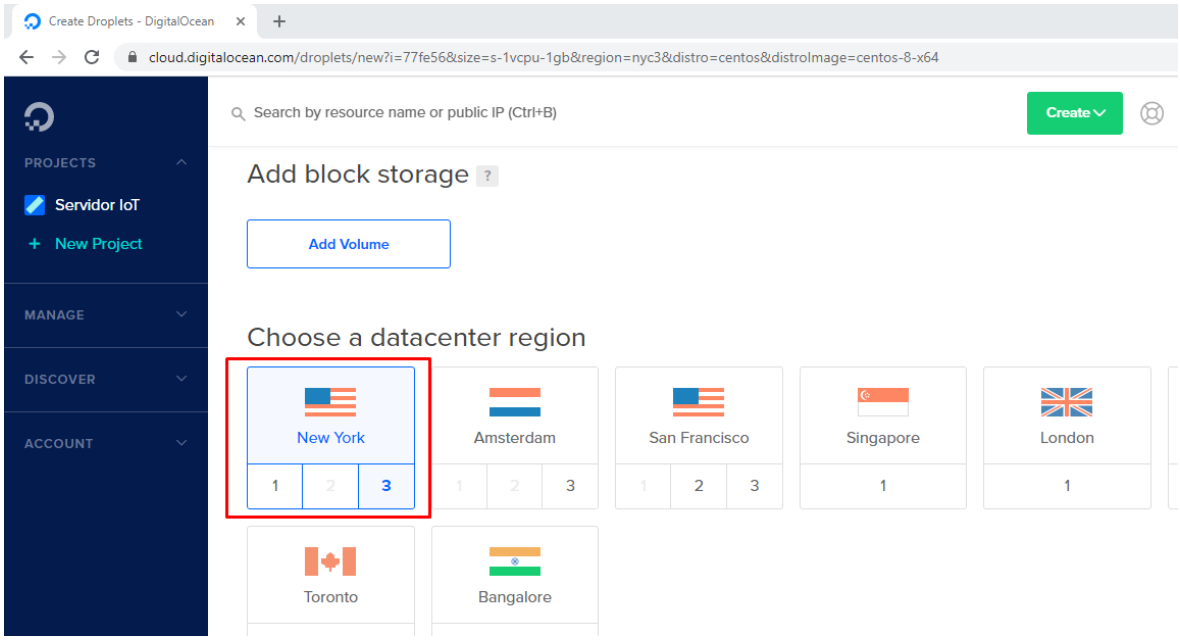

Figura 80. Selección de sitio de servidor físico (Elaboración propia de captura).

Una vez configurado los parámetros de creación del servidor, se procede con la configuración del servidor mediante comandos en la distro GNU/LINUX que se instaló en la nube, para ello se puede ingresar mediante protocolo SSH con la herramienta de conexión Putty que se descarga de internet.

Para el ingreso se debe conocer la IP publica que se asigna al servidor en la plataforma como se puede ver en la figura 81.

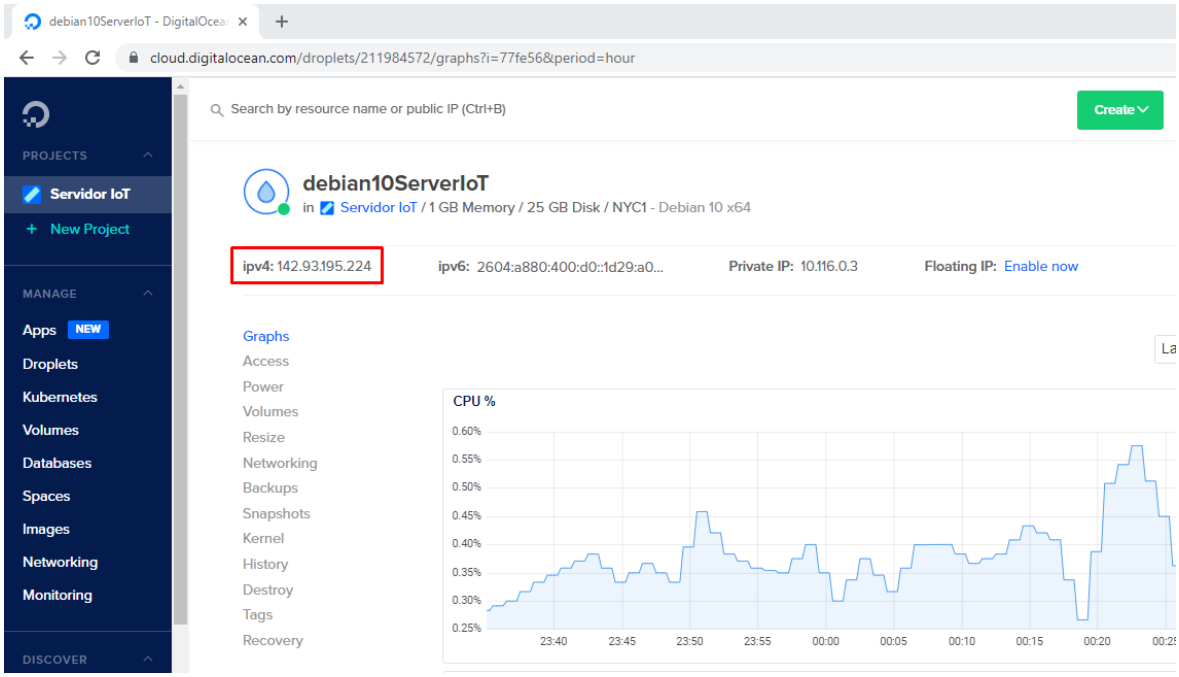

Figura 81. IP publica asignada al servidor (Elaboración propia de captura)

Una conocido la IP del servidor se procede con la conexión mediante una herramienta de administración por SSH que puede ser Putty o MobaXterm, y luego proceder a actualizarlo.

Lo primero que se procede a realizar es la actualización del sistema de archivos del sistema operativo mediante los comandos sudo apt-get update y sudo apt-get upgrade como se muestra en la figura 82.

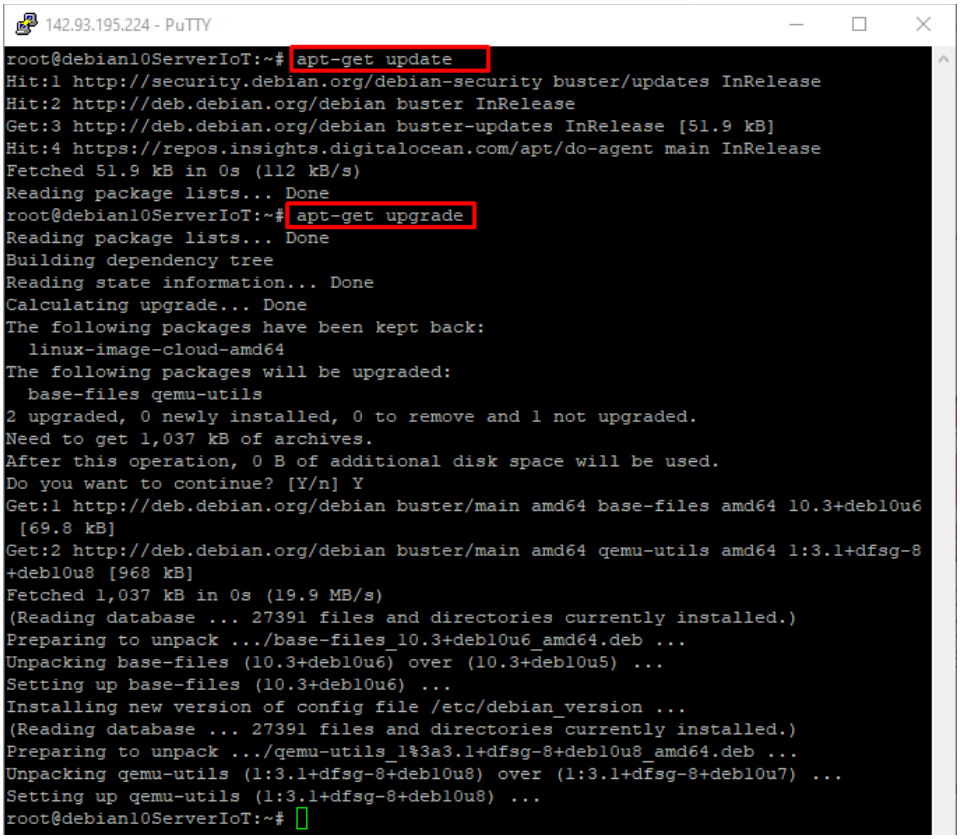

Figura 82. Actualización del servidor (Elaboración propia de captura).

3.2.4.2 Instalación de Apache

Para la instalación de apache se ejecuta la siguiente línea de comandos ***sudo apt install apache2 apache2-utils*** como se puede ver en la figura 83. Una vez terminado la instalación se ejecuta la siguiente línea de comando para reiniciar el servicio de Apache2 y se ejecuta la línea de comando ***sudo systemctl enable apache2*** para que el servicio se ejecute automáticamente al momento de iniciar el sistema. Una vez realizado el reinicio se verifica que el servicio se encuentre activo como se muestra en la figura 84.

Finalmente se configura www-data como usuario de apache de la raíz web, por defecto se encuentra como root. Para el cambio se usa el comando ***chown www-data:www-data /var/www/html/ -R.***

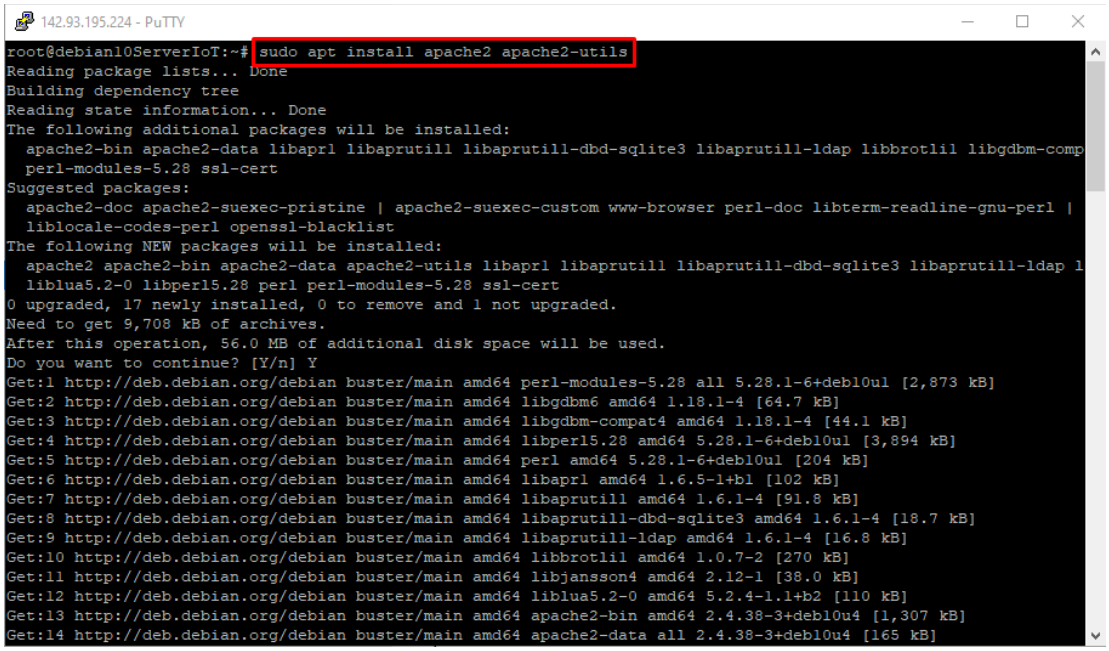

Figura 83. Línea de comando para la instalación de Apache (Elaboración propia).

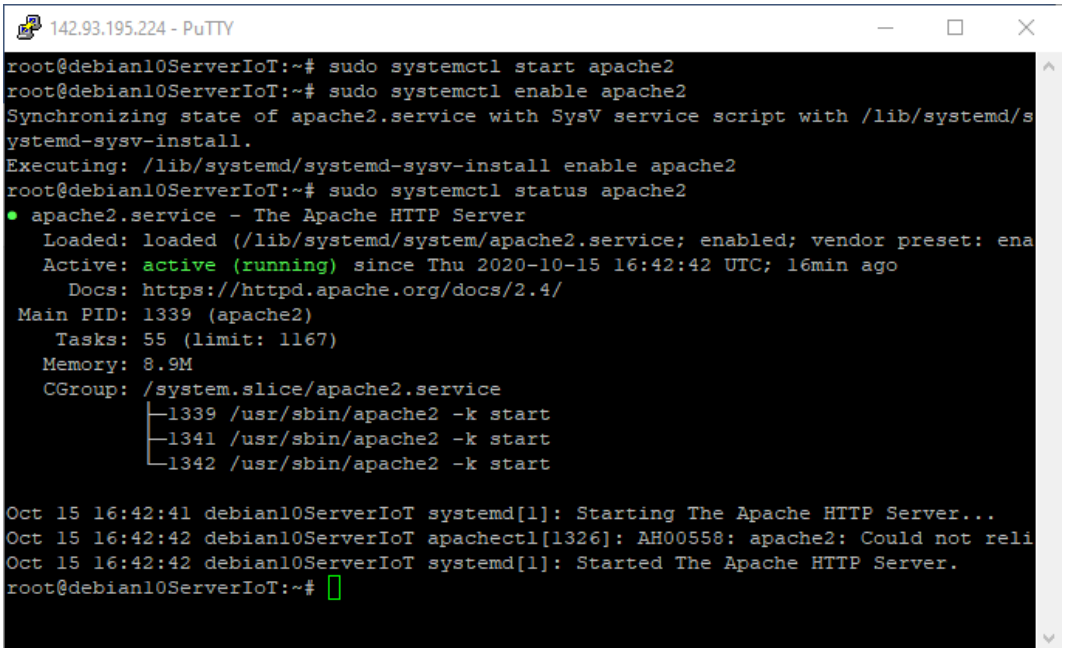

Figura 84. Reinicio y estado de Apache (Elaboración propia).

3.2.4.3 Instalación de base de datos Mariadb

Para la instalación de Mariadb se procede con la ejecución de la línea de comando ***sudo apt install mariadb-server mariadb-client,*** luego de la instalación se comprueba el estado del servicio con la línea de comandos ***systemctl status mariadb.service*** y se habilita para

que inicie cuando el sistema operativo se encienda con la línea de comando ***sudo systemctl enable mariadb***, como se muestra en la figura 85.

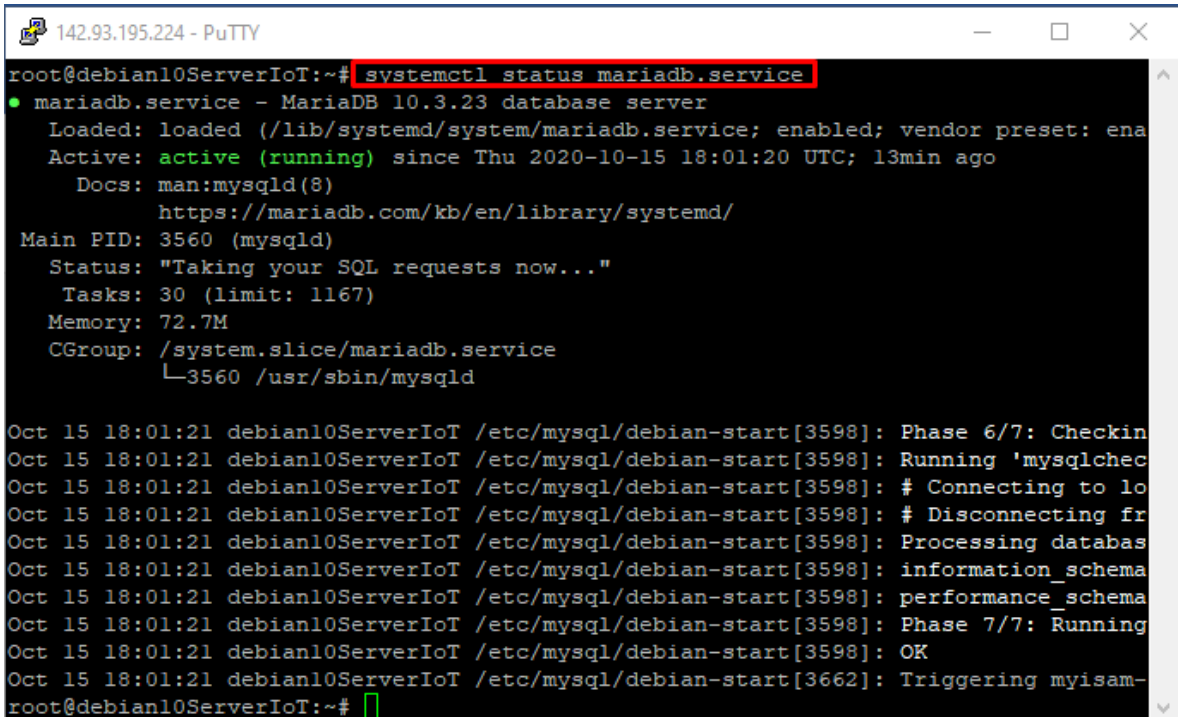

Figura 85. Estado del servicio de mariadb (Elaboración propia).

Para asegurar la instalación se procede a ejecutar la línea de comando ***sudo mysql_secure_installation,*** una vez que se ejecute el comando se solicitara configurar la contraseña de root, la eliminación de usuarios anónimos, deshabilitar el inicio de sesión remoto para el usuario root y finalmente borrar la base de datos demo y los accesos.

3.2.4.4 Instalación de php7.3

Para la instalación de php se procederá con la instalación de la versión 7.3 que aporta con una mejora en el rendimiento con respecto a las versiones anteriores. Para instalar php se ejecuta la línea de comandos siguiente ***sudo apt install php7.3 libapache2-mod-php7.3 php7.3-mysql php-common php7.3-cli php7.3-common php7.3-json php7.3-opcache php7.3-readline***. Luego de la instalación se procede con la habitación del módulo de apache php7.3 con la línea de comando ***sudo a2enmod php7.3,*** se reinicia el servicio de apache2 y finalmente se reinicia el sistema.

3.2.4.5 Instalación de phpMyAdmin

phpMyAdmin es una herramienta de administración grafica de la base de datos MySQL, para la instalación lo primero que se hará es verificar la instalación de los módulos necesarios que se requiere para phpMyAdmin. Los módulos que deben estar instalados son los siguientes:

- ✓ ***php-mbstring*** es una extensión de PHP utilizada para administrar cadenas no ASCII y convertir cadenas a diferentes codificaciones.
- ✓ ***php-zip*** es un módulo PHP que admite la carga de archivos .zip a phpMyAdmin
- ✓ ***php-gd*** es otro módulo PHP, este habilita el soporte para la biblioteca de gráficos GD.

Para la instalación se ejecutará la siguiente línea de comandos ***sudo apt install php-mbstring php-zip php-gd***, una vez instalado los módulos necesarios se procede con la descargar de phpMyAdmin, se revisa en la página principal de phpMyAdmin la última versión actualizada que en este caso es la 5.0.4. Para la descargar se utiliza la siguiente línea de comando con el gestor de descargas ***wget*** *https://files.phpmyadmin.net/phpMyAdmin/5.0.4/phpMyAdmin-5.0.4-all-languages.tar.gz*, luego de descargar se descomprime con el comando tar como se puede ver en la línea de comandos ***tar xvf phpMyAdmin-4.9.7-all-languages.tar.gz***, después de la descompresión se creara varios directorios y archivos dentro de la carpeta principal con el nombre phpMyAdmin-5.0.4-all-languages. La carpeta descomprimida se procede a mover hacia el directorio /usr/share/phpMyAdmin mediante la línea de comandos ***sudo mv phpMyAdmin-4.9.7-all-languages/ /usr/share/phpmyadmin.***

La configuración de phpMyAdmin se lo hará manualmente, se comenzará por crear una carpeta en la que se almacenará los archivos temporales que se vayan creando. La carpeta se crea mediante la línea de comandos ***sudo mkdir -p /var/lib/phpmyadmin/tmp***, luego se establece a www-data como perfil de usuario del sistema operativo, este perfil utiliza los servidores web apache de forma predeterminada, para esto se ejecuta la línea de comandos ***sudo chown -R www-data:www-data /var/lib/phpmyadmin.***

Luego se procede con la configuración del archivo config.sample.inc.php que se encuentra dentro del directorio /usr/share/phpmyadmin/, primero se procede a crear una copia del archivo con el nombre config.inc.php mediante la línea de comandos ***sudo cp /usr/share/phpmyadmin/config.sample.inc.php***

/usr/share/phpmyadmin/config.inc.php. Una vez creado la copia se procede a editar el archivo con la línea de comandos ***sudo nano /usr/share/phpmyadmin/config.inc.php***.

Dentro del archivo en la sección $cfg['blowfish_secret'] = 'cadenaCaracteres32Aleatorios'; se introduce una cadena de caracteres aleatorios, que no se requiere recordar, se usa internamente.

Luego en la sección **/* User used to manipulate with storage */**, se configura un usuario y password en el controluser y controlpass, luego en la sección **/* Storage database and tables */**, se des-comenta cada línea de esta sección para definir el almacenamiento de la configuración de phpmyadmin.

Por último, en la parte final del archivo se añade la línea **$cfg['TempDir'] = '/var/lib/phpmyadmin/tmp';** esta línea configura para que phpmyadmin pueda usar el directorio **/var/lib/phpmyadmin/tmp** como directorio temporal.

Una vez terminado la instalación se puede proceder con la configuración de usuarios para la administración de la base de datos, de la misma manera que se lo hace en phpmyadmin de la raspberry pi.

3.2.5 Instalación servicio FTP para transferencia de Archivos

En la plataforma IoT se instalará el servicio FTP que será de ayuda para la transferencia de archivos que serán necesarios para la configuración de la plataforma. Para la instalación del servicio ftp se procede a ejecutar la siguiente línea de comandos ***sudo apt-get install vsftpd*** como se observa en la figura 86, una vez instalado el servicio se procede con la configuración del archivo vsftpd.config para habilitar la escritura de los archivos a los usuarios como se muestra en la figura 87.

local_enable=YES

write_enable=YES

Luego se procede con el reinicio del servicio ejecutando la línea de comandos ***sudo service vsftpd restart***, después del reinicio se debe crear un usuario para el servicio mediante el comando ***sudo adduser useriot***, al momento de crear el usuario se pide la creación de una contraseña y algunos datos adicionales como se observa en la figura 88.

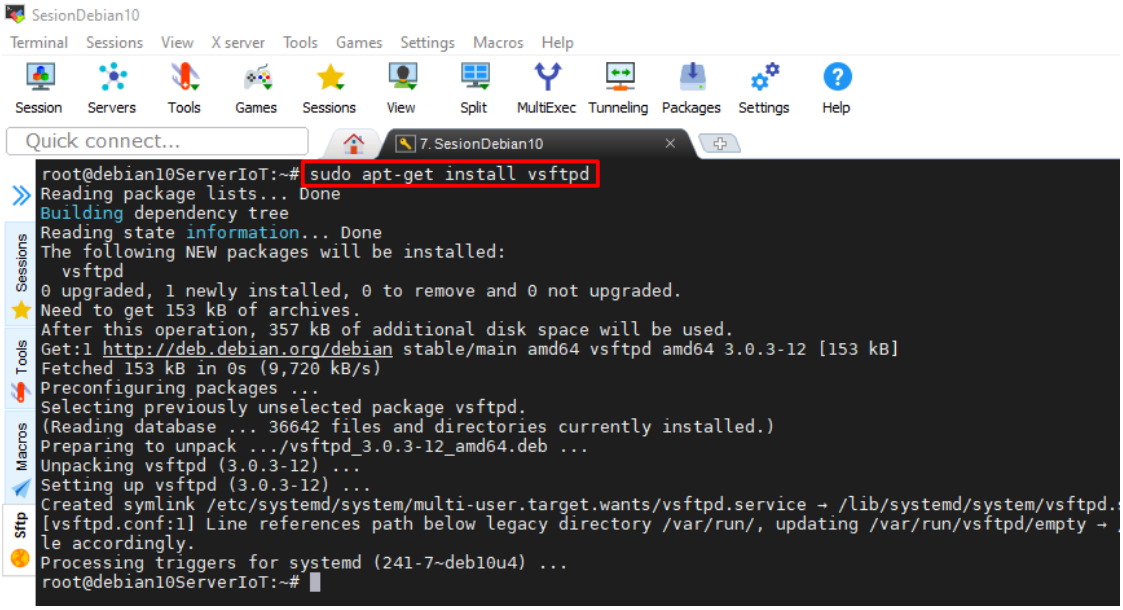

Figura 86. Instalación del servicio ftp (Elaboración propia).

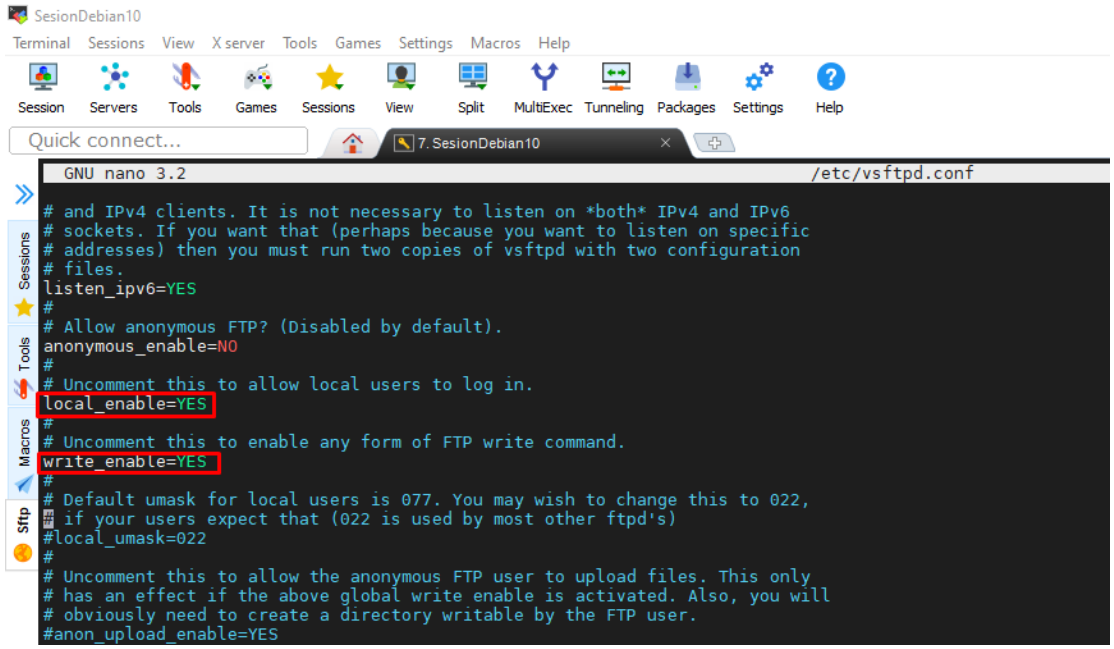

Figura 87. Edición del archivo vsftpd.conf (Elaboración propia).

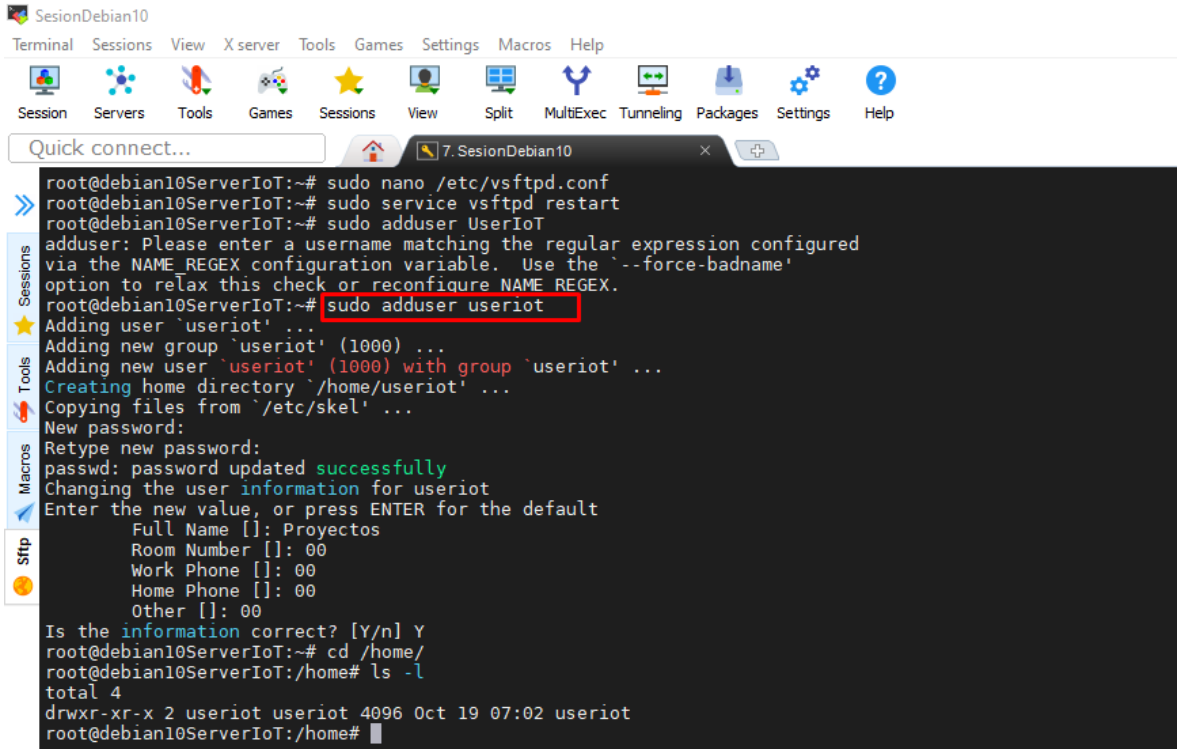

Figura 88. Creación del usuario en el servicio ftp (Elaboración propia).

Se debe crear una carpeta dentro del usuario configurado y darle los respectivos permisos de lectura y escritura. Esta carpeta contendrá los archivos de transferencia que serán útiles para el servidor, esta carpeta debe ser creada dentro del usuario 'useriot' creado, ya que los permisos que se debe otorgar son para este usuario creado.

Para comprobar el servicio se procede a realizar la conexión con FileZilla Client, como su nombre lo indica es un cliente mediante el cual se tiene la conexión al servicio en el host instalado. En la figura 89 se muestra la conexión hacia el servidor y la carpeta creada para la transferencia de archivos, se puede mostrar una conexión sin problemas.

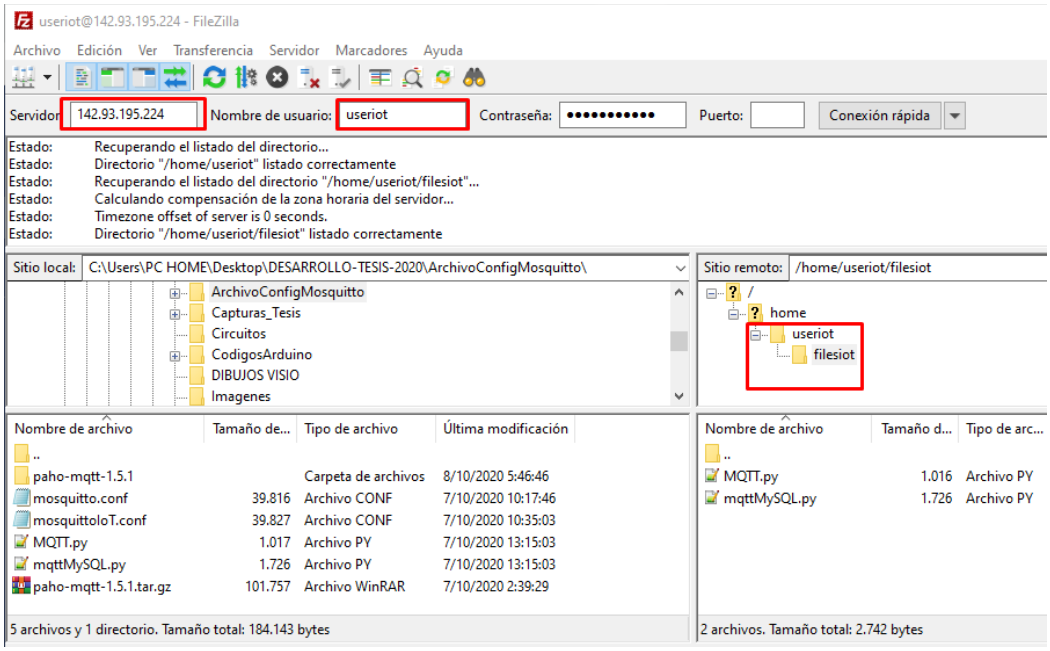

Figura 89. Conexión hacia el servidor (Elaboración propia).

3.2.6 Instalación del Broker y Cliente Mosquitto

Mosquitto es un paquete de software que permitirá instalar un bróker MQTT, este bróker trabaja con el protocolo de comunicación MQTT que actualmente es uno de los más utilizados para configurar una plataforma del Internet de las Cosas.

La instalación del bróker MQTT se va a realizar en el servidor de la nube, este contiene el sistema operativo GNU/Linux con la distribución de Debian_10. Para instalar el bróker, se instalará Mosquitto que es un mediador de mensajes que trabaja con el protocolo MQTT y que estará en constante disponibilidad.

Para la instalación se debe seguir una serie de procesos que permitirá una óptima instalación. Se comienza por la descargar de la clave de firma de mosquitto, para esto se ejecutará la línea de comandos que se describe a continuación: ***sudo wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key*** como se muestra en la figura 90.

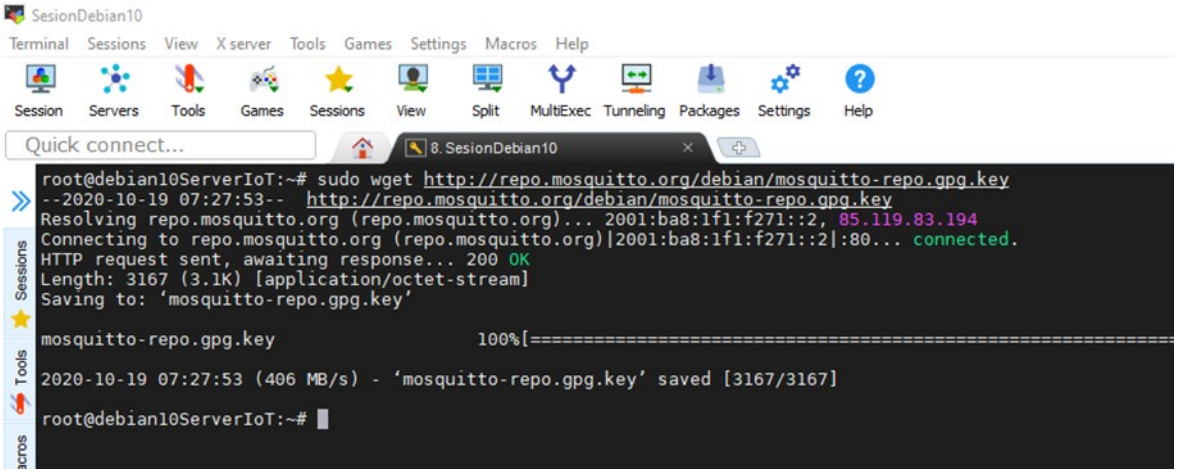

Figura 90. Descarga de la clave de firma de mosquitto (Elaboración propia).

Se añade la clave a una lista para autenticar el paquete que se va a descargar, para esto se ejecuta la línea de comandos ***sudo apt-key add mosquitto-repo.gpg.key*** como se muestra en la figura 91.

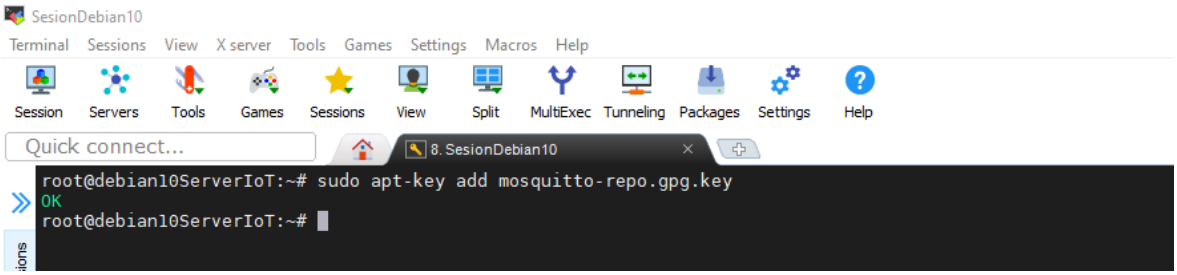

Figura 91. Añadir la clave en la lista para autentica r el paquete a descargar (Elaboración propia).

Luego se dirige a la ubicación ***cd /etc/apt/sources.list.d/*** para descargar los paquetes de instalación. Antes de la descarga se verifica la versión del sistema operativo instalado en el servidor con el comando ***cat /etc/os-release*** o a su vez con el comando ***lsb-release -a***. Como se muestra en la figura 92.

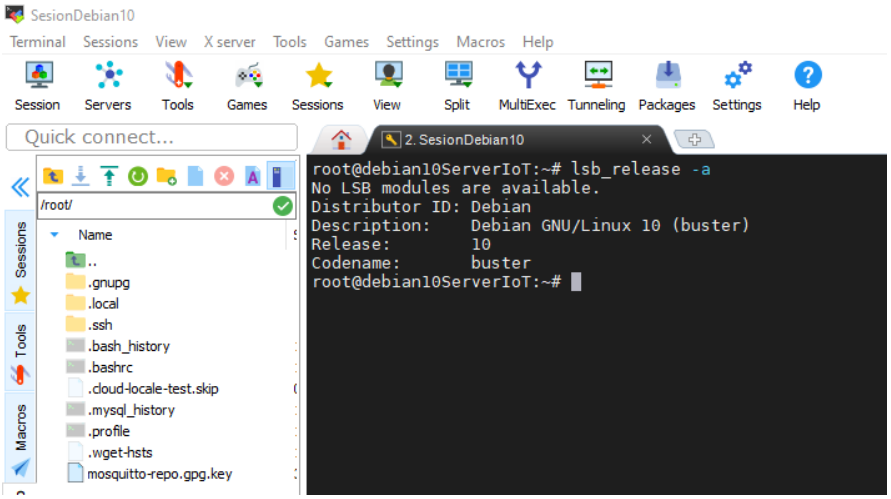

Figura 92. Versión del sistema operativo (Elaboración propia).

Se descarga el software para la instalación en el directorio indicado anteriormente con el comando ***sudo wget http://repo.mosquitto.org/debian/mosquitto-buster.list*** como se ve en la figura 93, la descarga se realiza desde el repositorio de mosquitto para luego ingresar como root con el comando ***sudo -i*** y proceder con la instalación.

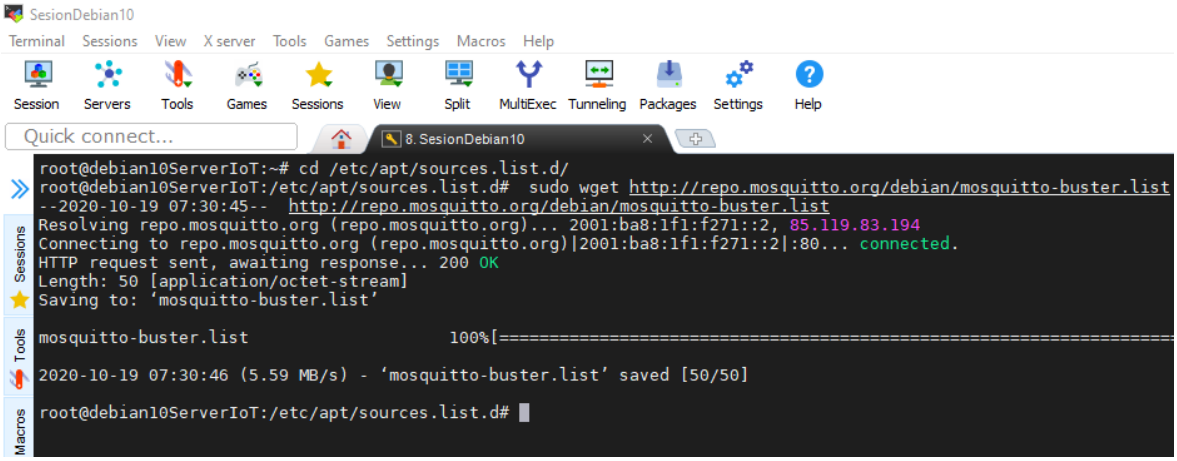

Figura 93. Descarga de software desde el repositorio de mosquitto (Elaboración propia).

Luego, instala el bróker mosquito como root, para lo cual se ejecuta el comando ***apt-get install mosquitto*** como se muestra en la figura 94.

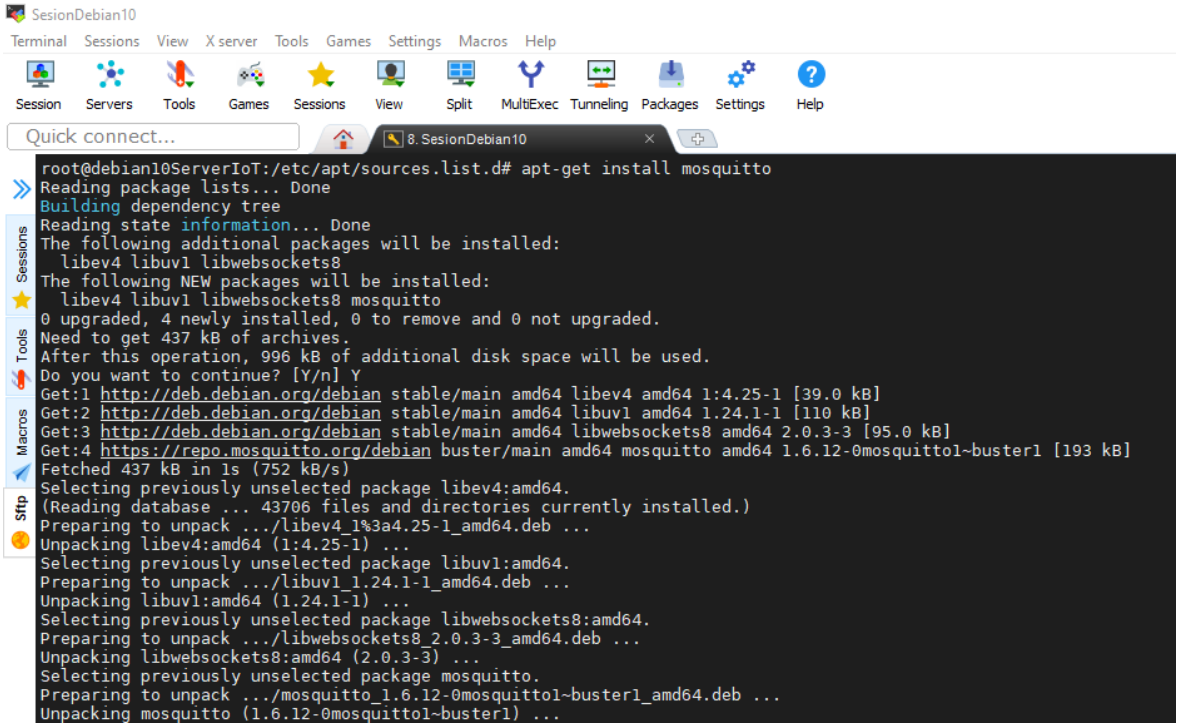

Figura 94. Instalación del broker Mosquitto (Elaboración propia).

Ahora se instalará el cliente Mosquitto para verificar el funcionamiento de del bróker en la publicación y suscripción a un topic, para la instalación del cliente se ejecuta el comando ***apt-get install mosquitto-clients***, la ejecución del comando y la instalación del cliente se muestra en la figura 95.

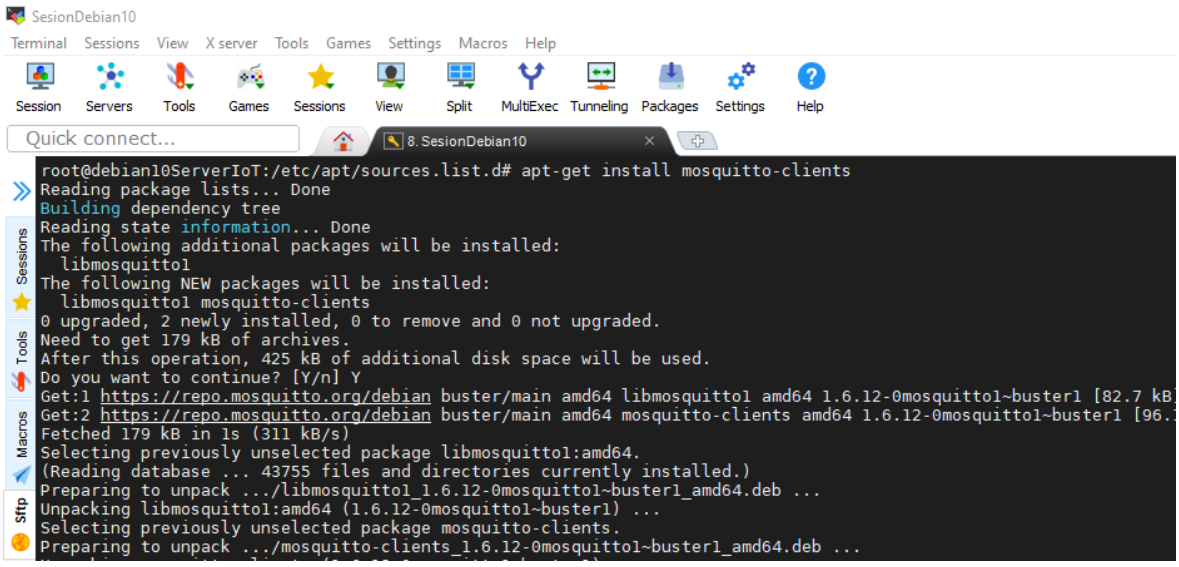

Figura 95. Instalación del cliente Mosquitto (Elaboración propia).

3.2.7 Instalación de Python y las librerías necesarias.

Se procede con la instalación de Python 3 que es la versión más reciente, se debe verificar en el sistema operativo GNU/LINUX la instalación del software ya que por default viene instalado, es necesario saber la versión de Python que contiene el sistema operativo porque la versión 2 tiene librerías obsoletas que no permite la instalación y se tiene problemas para corres los archivos ***files.py***.

La primera librería que se utilizara es la de python3-paho-mqtt, es una librería que implementa versiones 5.0, 3.1.1 y 3.1 del protocolo MQTT. Proporciona una clase de cliente que permite conectarse a un agente MQTT para publicar, suscribirse a temas y recibir los mensajes publicados en el bróker.

Para la instalación se procede con la ejecución de la línea de comandos ***sudo apt-get install python3-paho-mqtt*** como se observa en la figura 96.

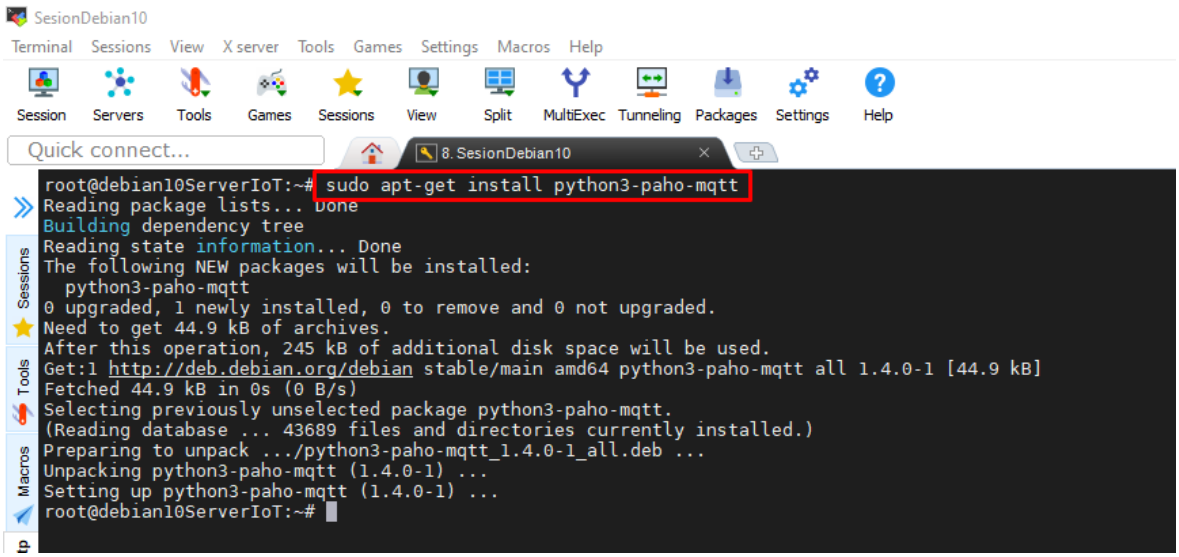

Figura 96. Instalación de librería paho-mqtt (elaboración propia).

La segunda librería que se utilizara es Python-MySQL, es una librería que permite un interfaz entre el servidor de base de datos y Python, para poder almacenar la información recibida e ingresar a la base de datos en Tablas. Para la instalación de la librería se ejecuta la línea de comando ***sudo apt-get install python-mysqldb***, la ejecución del comando se muestra en la figura 97.

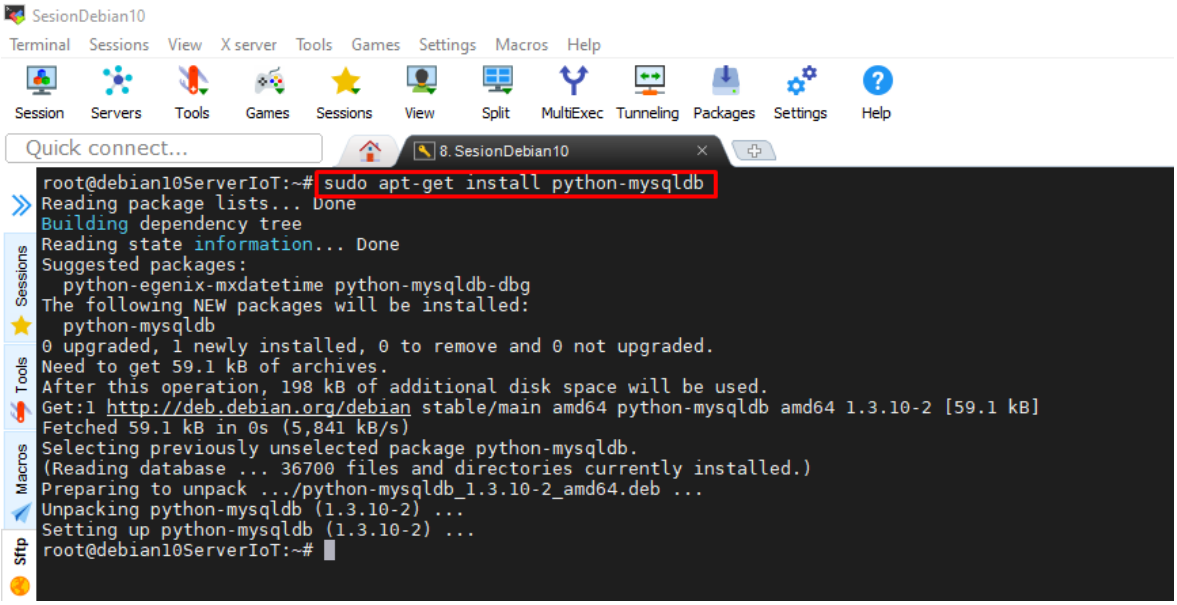

Figura 97. Instalación librería python-mysqldb (Elaboración propia).

Finalmente, se instalará una librería que no es muy importante, su función es instalar paquetes para Python, al ejecutar una instalación se usa pip acompañado del paquete que se requiere instalar. Para instalar pip se ejecuta la línea de comando ***sudo apt-get install python3-pip***, la instalación se observa en la figura 98.

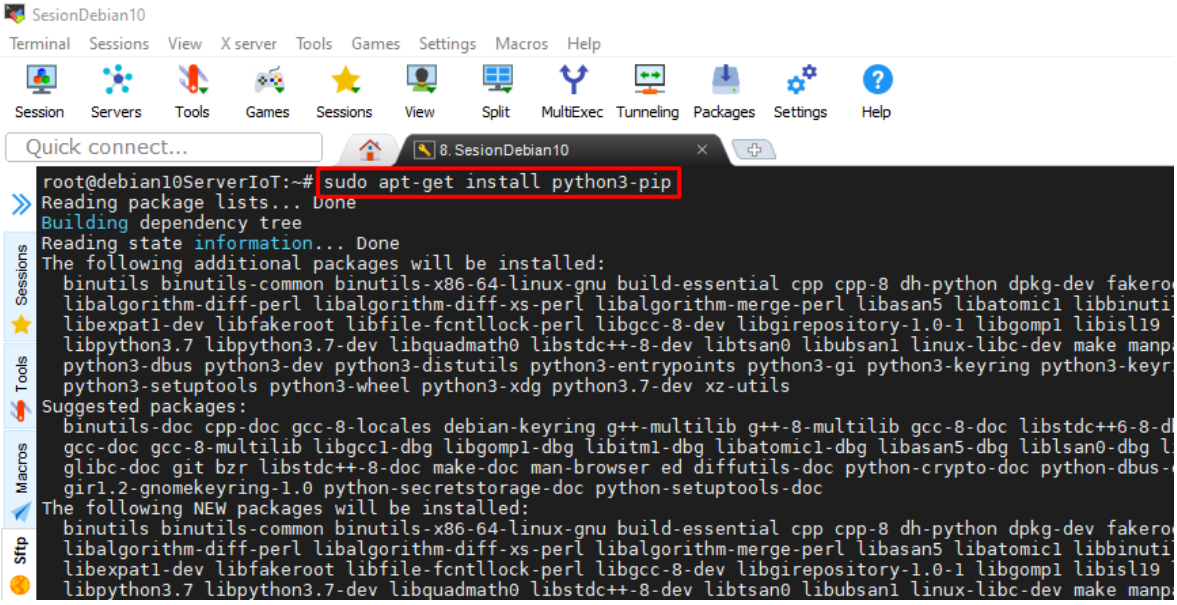

Figura 98. Instalación del instalador de paquetes para python3 (Elaboración propia).

3.2.8 Envío de datos desde Arduino mediante protocolo MQTT

Para él envió de datos desde los sensores conectados a la tarjeta Arduino se procede con la instalación de una librería para el IDE de Arduino denominada PubSubClient, esta librería permite enviar y recibir mensajes MQTT, soporta la versión 3.1.1 del protocolo MQTT y adicional puede ser configurado una versión 3.1 si se requiere. La librería se muestra en la figura 99.

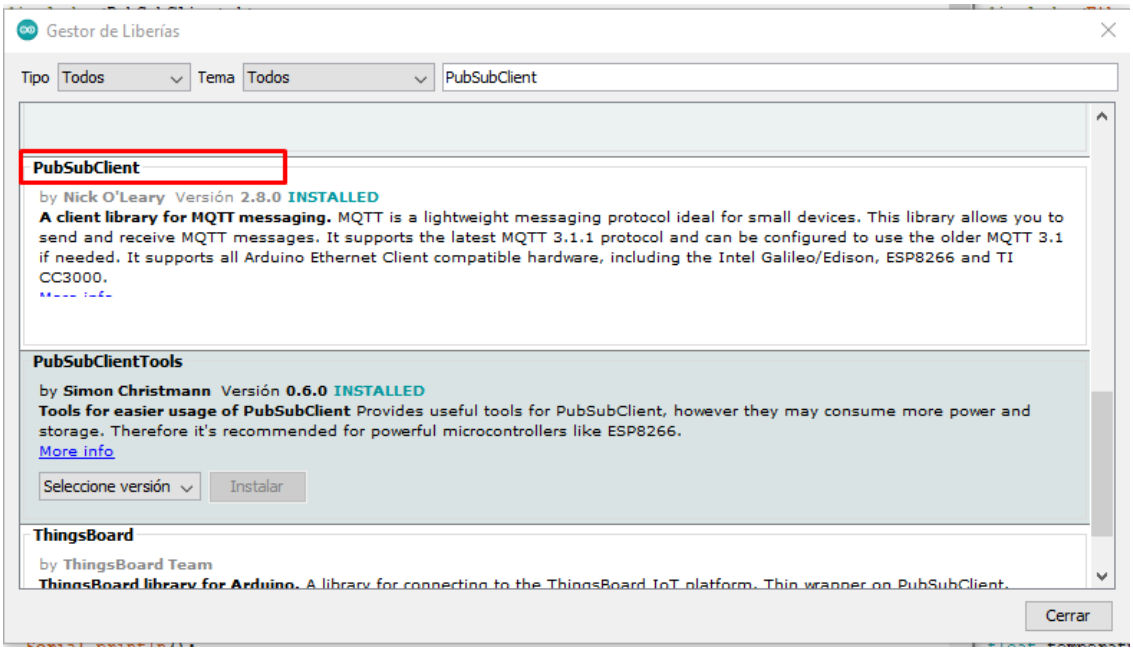

Figura 99. Librería de Arduino para envió de datos por MQTT *(Arduino, 2020)*

Luego de instalar la librería se procede con la programación para el envío de información al broker, dentro del código de programación se debe añadir la librería para poder usar cada una de las funciones que permitan la transferencia de datos hacia el bróker mediante una publicación al topic que se suscribe. El código de programación se encuentra descrito en el Anexo 3. Este código posteriormente se dividirá para que una tarjeta trabaje con publicación de mensajes y la otra tarjeta con suscripción a topics.

Dentro del código se describe las funciones importantes que se utiliza para poder transmitir la información de los sensores hacia el broker MQTT, lo primero que se debe hacer es incluir la librería <PubSubClient.h> como se puede ver en la figura 100. De la misma manera se puede observar que el siguiente paso es la creación de una instancia parcialmente inicializada, esto quiere decir que se debe configurar los parámetros del servidor antes de utilizar.

MQTTCloudRev01 Arduino 1.8.13 (Windows Store 1.8.42.0)

Archivo Editar Programa Herramientas Ayuda

MQTTCloudRev01 §

```
#include <DHT.h>          //Libreria para sensor Temperatura
#include <DHT_U.h>        //Libreria para sensor Temperatura
#include <SPI.h>          //Libreria para shield Ethernet
#include <Ethernet.h>     //Libreria para shield Ethernet
#include <PubSubClient.h> //Libreria para Publicar y Suscribir con MQTT

byte mac[] = {0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED}; //Configuracion de direccion MAC para Arduino
IPAddress ip(10 , 10, 70, 6);               //Configuracion de direccion IP para Arduino
IPAddress server(142 ,93 ,195 , 224);           //IP del servidor a conectarse

#define PINDHT 2              //Se define el PIN 22 como ingreso de datos ARDUINO MEGA
#define DHTTYPE DHT22         //Se define el tipo de sensor

DHT sdht(PINDHT, DHTTYPE);      //Inicializa la variable con el PIN y tipo de sensor
EthernetClient ethClient;       //Se configura la variable para Cliente Ethernet
PubSubClient client(ethClient); //Se crea una instancia de cliente parcialmente inicializada
```

Figura 100. Incluir librería y creación de una instancia (Elaboración propia).

Luego se crea los topics (temas), donde se va a direccionar toda la información de cada uno de los sensores y actuadores que se crea en el sistema IoT, los topics en el código de programación se muestra en la figura 101.

MQTTCloudRev01 Arduino 1.8.13 (Windows Store 1.8.42.0)

Archivo Editar Programa Herramientas Ayuda

MQTTCloudRev01 §

```
int F2 = 4;                   //Se define el PIN para detector de fase 2
char temp[6];                 //Se crea Buffer para alamacenar string
char hume[6];
char tx_A[6];
char tx_B[6];
const char* topicPub1 = "/proyectoIoT/estacion/temperatura";   //Topic temperatura
const char* topicPub2 = "/proyectoIoT/estacion/humedad";       //Topic para humedad
const char* topicPub3 = "/proyectoIoT/estacion/transmisorUno"; //Topic transmisor1
const char* topicPub4 = "/proyectoIoT/estacion/transmisorDos"; //Topic transmisor2
const char* topicPub5 = "/proyectoIoT/estacion/redelectrica";  //Topic redelectrica
const char* topicSub = "/proyectoIoT/estacion/reinicioEquipo"; //Topic para Publicacion

void callback(char* topic, byte* payload, unsigned int length)
{
  Serial.print("Llego el mensaje [");
  Serial.print(topic);
```

Figura 101. Creación de los topics en el código de programación (Elaboración propia).

Se crea una función de devolución de llamada de suscripción que se utiliza para suscribir a topics al cliente. Esta función se llama cuando llegan nuevos mensajes al cliente. La función de llamada se muestra en la figura 102.

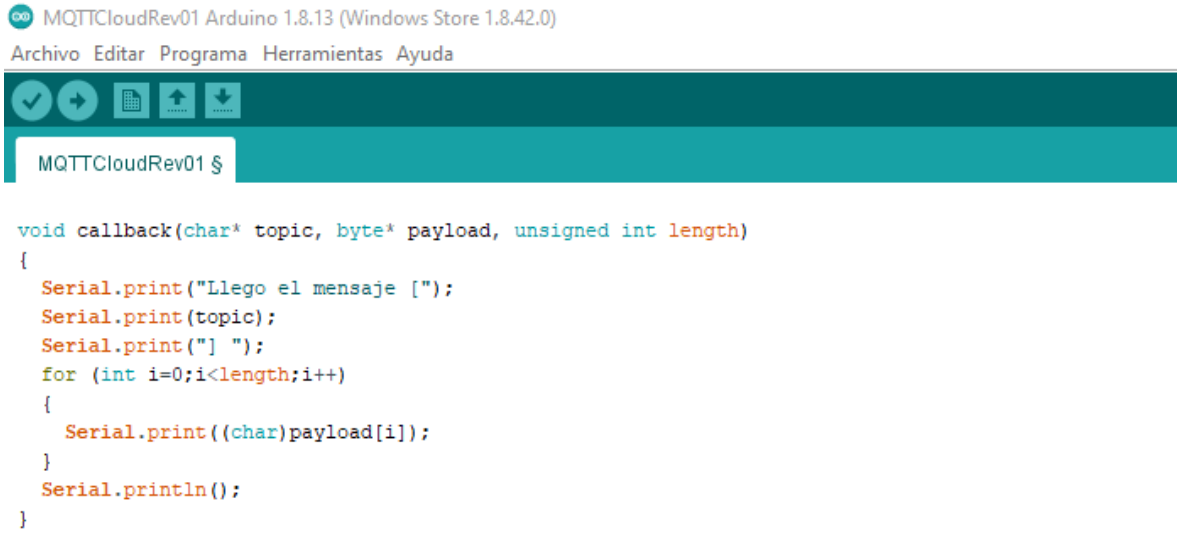

Figura 102. Función de devolución de llamada (Elaboración propia).

Se procede a configurar los parámetros del servidor como la IP o dirección de web del servidor, puerto de comunicaciones. Adicional se carga la función de devolución de llamada como se observa en la figura 103.

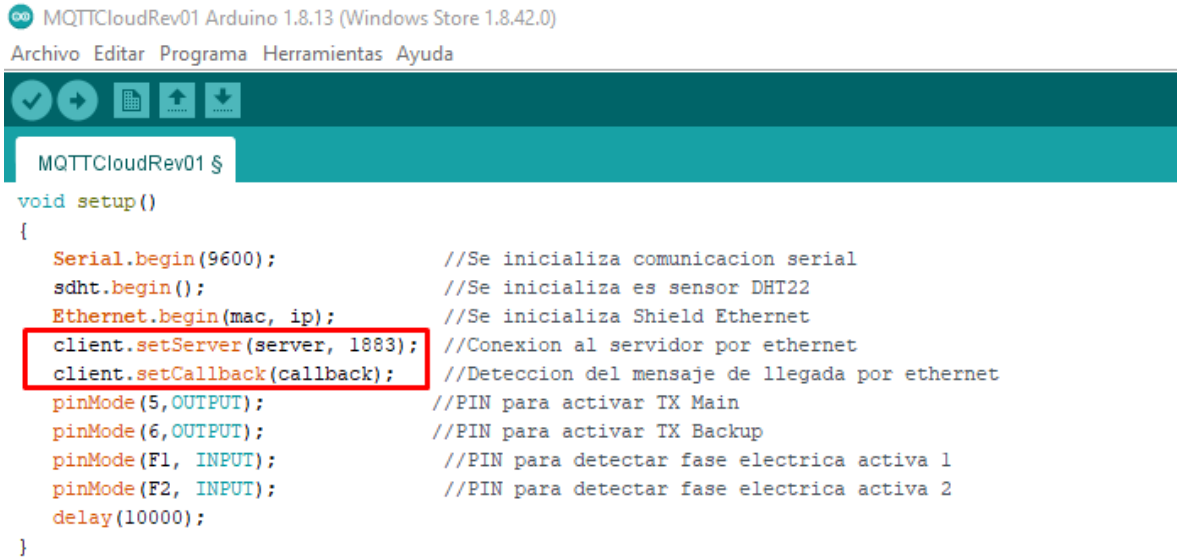

Figura 103. Configurar parámetros del servidor (Elaboración propia).

Finalmente se crea una condición para que se ejecute las funciones mientras esta sea verdadera, para la publicación y suscripción en los topics se usa las funciones client.publish() y client.subscribe(), en estas funciones se carga el topic a conectarse y el parámetro a publicar como se observa en la figura 104.

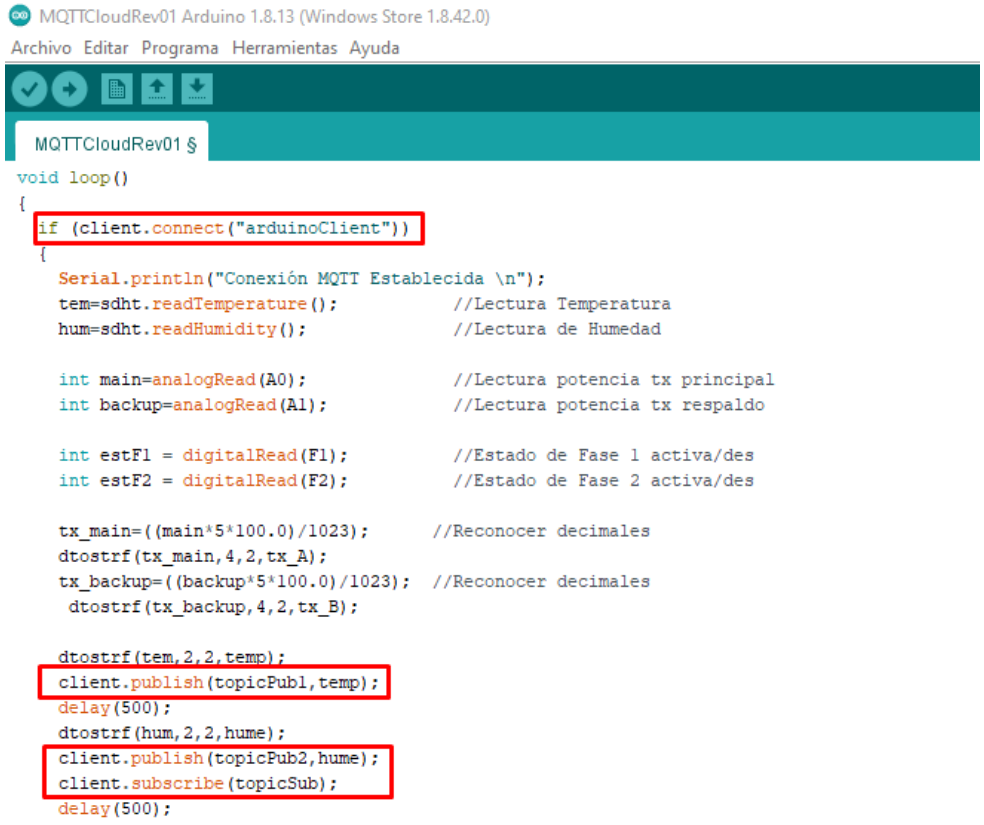

Figura 104. Publicación y suscripción a los topics.

3.2.9 Envío de información hacia la base de datos con Python

Para él envió de los datos obtenidos de los sensores hacia la base de datos MariaDb se procede con la creación de un script de Python con el nombre mqttMySQL.py, este archivo se encuentra en la ubicación **/home/useriot/filesiot**, y permite leer los datos direccionados al Broker MQTT mediante librerías de Python instaladas previamente. La librería es **python-mysqldb,** esta librería lee los datos y direcciona los datos hacia las tablas creadas en la base de datos.

Estas tablas se crean de acuerdo con la información contenida en la Tabla 8, en esta tabla se muestra los datos que se cargaran en cada una de las columnas de la tabla, para mantener un registro completo de los parámetros de la estacion a monitorear.

Nombre Base de Datos: estaciondb			
Nombre de Tabla	**Columna Uno**	**Columna dos**	**Columna tres**
temperatura	id	grados_C	fecha
transmisorUno	id	Potencia[W]	fecha
transmisorDos	id	Potencia[W]	fecha
estadoRedElectrica	id	Status	fecha
humedad	id	porcentaje [%]	fecha
actuadorUno	id	Estado	fecha
actuadorDos	id	Estado	fecha
reinicioSistema	id	Estado	fecha

Tabla 8. Información a cargar en cada tabla (Elaboración propia)

Para el envío de información a cada una de las tablas se debe crear un script de Python que lea toda la información enviada por los clientes hacia el broker y luego envié cada dato a la tabla correspondiente en la base de datos, y así almacenar para que se encuentre disponible.

Dentro del script de Python, lo primero que se debe hacer es incluir las dos librerías instaladas anteriormente y las necesarias para crear el código de programación como se muestra en la figura 105.

```
  GNU nano 3.2                                                    mqttMySQL.py

#SCRIPT PYTHON LECTURA Y ENVIO DE DATOS A TABLAS EN BASE DE DATOS
import paho.mqtt.client as mqtt
import sys
import MySQLdb

#Abrir conexion con base de datos
try:
    db = MySQLdb.connect("127.0.0.1", "UserIoT", "user-iot", "estaciondb")
except:
    print("No se puede conectar a base de datos")
    print("Cerrando...")
    sys.exit()

#Creacion de cursor
cursor = db.cursor()
```

Figura 105. Incluir las librerías necesarias (Elaboración propia).

La librería ***paho.mqtt.client***, es un cliente que permite suscribirse a cualquier topic que se configura más adelante en el código, la librería ***MySQLdb,*** permite enviar el dato recibido en el topic suscrito hacia una de las tablas en la base de datos.

Para abrir la conexión a la base de datos se define una variable db para almacenar la información en la base de datos y con la función MySQLdb.connect(), se carga esta información como el localhost, el usuario y password de ingreso a la base de datos, y el

nombre de la base de datos creada, adicional se imprime un mensaje en caso de no tener una conexión hacia la base de datos. Como se observa en la figura 106.

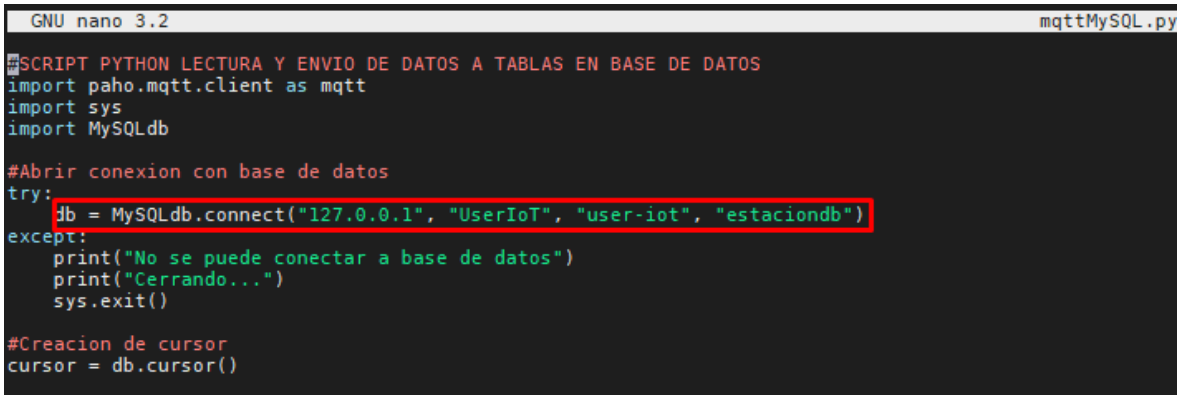

Figura 106. Ingreso de parámetros de conexión a la base de datos (Elaboración propia).

Luego se crea un cursor que se ubicara en la base de datos para cargar la información en las tablas, para ello se crea una variable llamada cursor con la función db.cursor como se observa en la figura 107.

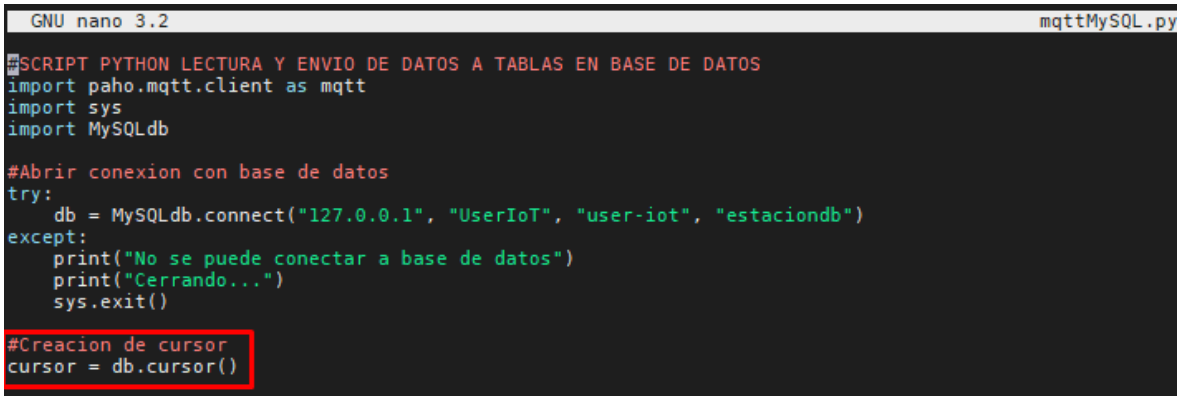

Figura 107. Creación del cursor (Elaboración propia).

Se crea dos Devoluciones de llamada que se ejecutaran cuando se realiza una conexión, la primera se define como on_connect(), que determina cuando el servidor recibe un paquete CONNECT bien formado, pero si el servidor no puede procesarlo por alguna razón; entonces el servidor envía un paquete CONNACK que contiene un código de retorno de conexión distinto de cero, caso contrario se cierra la conexión. Los códigos de retorno de conexión y su descripción se muestran en la tabla 9.

Valor	Respuesta de código de retorno	Descripción
0	Conexión 0x00 aceptada	Conexión aceptada
1	0x01 Conexión rechazada, versión de protocolo inaceptable	El servidor no admite el nivel del protocolo MQTT solicitado por el cliente.
2	0x02 Conexión rechazada, identificador rechazado	El identificador del cliente es UTF-8 correcto, pero el servidor no lo permite.
3	0x03 Conexión rechazada, servidor no disponible	Se ha realizado la conexión de red, pero el servicio MQTT no está disponible.
4	0x04 Conexión rechazada, nombre de usuario o contraseña incorrectos	Los datos del nombre de usuario o la contraseña están mal formados.
5	0x05 Conexión rechazada, no autorizada	El cliente no está autorizado a conectarse
6-255		Reservado para utilización futura.

Tabla 9. Valores del Código de retorno de conexión *(OASIS, 2015)*.

La segunda Devolución de llamada se define como on_message, esta se ejecuta una vez realizada la conexión cuando un mensaje de publicación es recibido por el servidor, esta función permite leer el mensaje contenido como string de datos, la información del dato recibido desde un sensor se lee en el str(msg.payload) que a su vez se va almacenar en la tabla de la base de datos correspondiente como se observa en la figura

```
#Devolucion de llamada cuando un el cliente recibe una respuesta de Conexion ACK del servidor.
def on_connect(client, userdata, flags, rc):
    print("Conectado - codigo de resultado: "+str(rc))

    #Topic a suscribirse por parte del cliente.
    client.subscribe("/proyectoIoT/estacion/#")

#Devolucion de llamada cuando un mensaje de publicacion es recibido del servidor.
def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload))

    lista = msg.topic.split("/")
    sql = """INSERT INTO `parametrosEstacion` (`id`,`lugar`,`sensor`,`valor`,`fecha`)
        VALUES (NULL, '"""+lista[2]+"""','"""+lista[3]+"""','"""+str(msg.payload)+"""', current_tim
    try:
        # Ejecutar un comando SQL
        cursor.execute(sql)
        db.commit()
        print("Guardando en base de datos....OK")
    except:
        db.rollback()
        print("Guardando en base de datos...fail")
```

Figura 108. Creación Devolución de llamadas y almacenamiento de datos (Elaboración propia).

Adicional, se crea mensajes que indiquen si el dato se guardó correctamente en la base de datos o no como se puede ver en la figura 108.

Para finalizar el código, se ejecutan las Devoluciones de llamada y luego se realiza la configuración de parámetros para conexión al servidor, aquí se configura la IP del servidor o dirección web, y el puerto mediante el cual se va a comunicar, adicional un tiempo de espera de conexión. Se imprime un mensaje comunicando si no se puede conectar al servidor por motivo de bloqueo de puertos o dirección IP invalida y luego la conexión se cierra como se puede ver en la figura 109.

```
client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

try:
    client.connect('142.93.195.224', 1883, 20)
except:
    print("No se puede connectar con el MQTT Broker...")
    print("Cerrado")
    db.close()
    sys.exit()

#client.username_pw_set("lqeamtbn", "vhOcU_Vcszxp")

try:
    client.loop_forever()
except KeyboardInterrupt: #precionar Ctrl + C para salir
    print("Cerrando...")
    db.close()
```

Figura 109. Configuración IP del servidor y puerto de comunicación (Elaboración propia).

En la figura 109 también se puede ver que se realiza un lazo de ejecución del programa, y adicional se configura una interrupción mediante “ctrl+C”, cuando se requiera detener la ejecución del programa.

Una prueba de la ejecución del código se puede ver en la figura 110, en la ejecución del código se muestra el resultado de conexión igual a cero, esto valor indica que la conexión fue aceptada e inmediatamente se inicia la captura de datos guardando cada dato en las tablas de la base de datos.

```
root@debian10ServerIoT:/home/useriot/filesiot# python mqttMySQL.py
Conectado - codigo de resultado: 0
/proyectoIoT/estacion/activarUno 0
Guardando en base de datos....OK
/proyectoIoT/estacion/activarB 0
Guardando en base de datos....OK
/proyectoIoT/estacion/temperatura 21.90
Guardando en base de datos....OK
/proyectoIoT/estacion/humedad 49.50
Guardando en base de datos....OK
/proyectoIoT/estacion/transmisorUno 87.98
Guardando en base de datos....OK
/proyectoIoT/estacion/transmisorDos 80.16
Guardando en base de datos....OK
/proyectoIoT/estacion/redelectrica FAIL
Guardando en base de datos...fail
```

Figura 110. Ejecución del script de Python (Elaboración propia).

Se debe crear un script de Python por cada sensor o cliente, para poder almacenar en tablas diferentes cada tipo de dato que se obtendrá de cada publicación que se realiza en el broker MQTT.

3.2.10 Ejecución del script de Python con el inicio del sistema operativo

Se debe crear un script de Python por cada cliente que se suscriba en el broker MQTT, este script debe ejecutarse automáticamente cada vez que el sistema operativo o el servidor se reinicie, para esto cada script debe arrancar como que fuera un servicio que se puede iniciar, detener o reiniciar mediante ejecución de comandos en Linux.

En la figura 111, se muestra dos carpetas; en la primera se muestra los archivos Python creados para cada los cinco topics de publicación y 3 topics de suscripción, de acuerdo a la descripción hecha en la sección 3.2.2. Mientras que en la carpeta dos se encuentra los scripts de servicios file.service creados para cada topic.

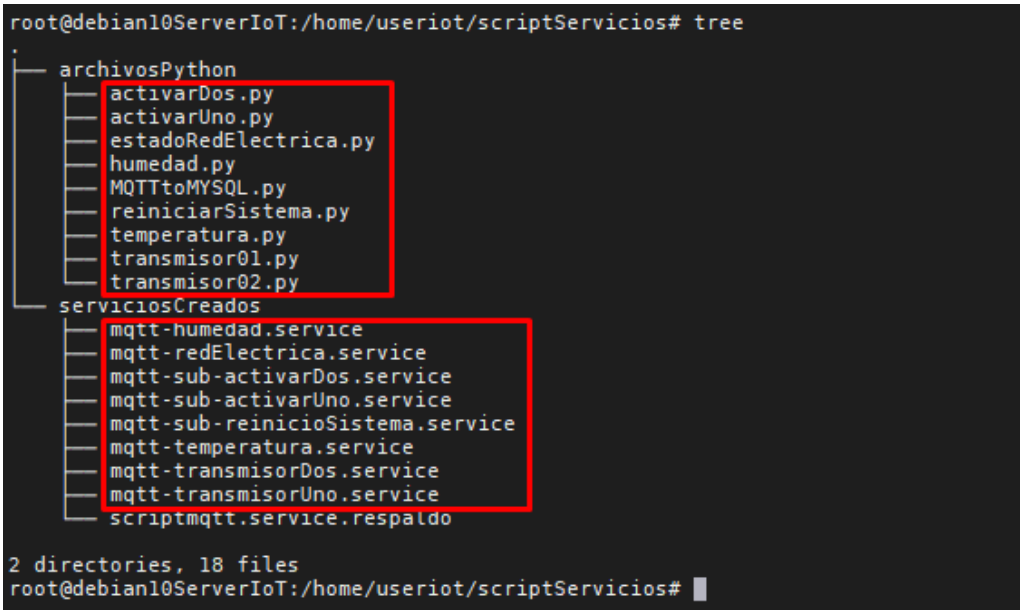

Figura 111. Archivos python y scripts.service creados (Elaboración propia).

Cada script de servicio se crea con extensión ***.service***, en el archivo creado se edita y se escribe la configuración mostrada en la figura 112, donde se configura el nombre del servicio a crear, el tipo de servicio y el directorio donde se encuentra ubicado el archivo de temperatura.py para el ejemplo.

```
  GNU nano 3.2                                                    mqt

[Unit]
Description=Servicio temperatura
After=multi-user.target

[Service]
Type=idle
ExecStart=/usr/bin/python /home/useriot/scriptServicios/archivosPython/temperatura.py

[Install]
WantedBy=multi-user.target
```

Figura 112. Contenido a configurar en los archivos .service (Elaboración propia).

Una vez creado cada uno de los scripts de servicio, se deben copiar al directorio ***/lib/systemd/system***, desde donde se ejecutan este tipo de archivos como se muestra en la figura 113.

```
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountall-bootclean.service -> /dev/null
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountall.service -> /dev/null
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountdevsubfs.service -> /dev/null
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountkernfs.service -> /dev/null
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountnfs-bootclean.service -> /dev/null
lrwxrwxrwx 1 root root    9 Apr 27  2020 mountnfs.service -> /dev/null
-rw-r--r-- 1 root root  201 Nov 12 20:31 mqtt-humedad.service
-rw-r--r-- 1 root root  228 Nov  1 10:49 mqtt-redElectrica.service
-rw-r--r-- 1 root root  208 Nov 12 20:33 mqtt-sub-activarDos.service
-rw-r--r-- 1 root root  208 Nov 12 20:33 mqtt-sub-activarUno.service
-rw-r--r-- 1 root root  219 Nov 12 20:33 mqtt-sub-reinicioSistema.service
-rw-r--r-- 1 root root  209 Nov  1 10:49 mqtt-temperatura.service
-rw-r--r-- 1 root root  220 Nov  1 10:49 mqtt-transmisorDos.service
-rw-r--r-- 1 root root  222 Nov  1 11:19 mqtt-transmisorUno.service
-rw-r--r-- 1 root root  532 Feb 14  2019 multi-user.target
drwxr-xr-x 2 root root 4096 Aug 30 12:00 multi-user.target.wants
-rw-r--r-- 1 root root  643 Aug 25  2018 networking.service
-rw-r--r-- 1 root root  505 Feb 14  2019 network-online.target
-rw-r--r-- 1 root root  502 Feb 14  2019 network-pre.target
-rw-r--r-- 1 root root  521 Feb 14  2019 network.target
-rw-r--r-- 1 root root  554 Feb 14  2019 nss-lookup.target
-rw-r--r-- 1 root root  513 Feb 14  2019 nss-user-lookup.target
-rw-r--r-- 1 root root  354 Mar 21  2019 ntp.service
-rw-r--r-- 1 root root  394 Feb 14  2019 paths.target
-rw-r--r-- 1 root root  155 Dec 17  2018 phpsessionclean.service
```

Figura 113. Archivos copiados a /lib/systemd/system (Elaboración propia).

Luego de copiar los archivos se reinicia el servidor, y finalmente se procede a verificar el estado, iniciar cada servicio, y habilitar mediante las líneas de comandos ***systemctl status mqtt-temperatura.service***, ***systemctl start mqtt-temperatura.service***, ***systemctl enable mqtt-temperatura.service***. Luego de realizar este procedimiento a cada uno de los servicios quedan habilitados para iniciar automáticamente con el sistema operativo.

3.2.11 Configuración de seguridad con Autenticación y Autorización.

El broker mosquitto se encuentra funcionando correctamente sin seguridad, esto quiere decir que cualquiera puede acceder por el puerto 1883 del servidor, suscribirse y publicar en un topic.

Se procederá a configurar un broker MQTT con autenticación para asegurar un poco el acceso y se pueda exponer el servidor configurado como público, así se tendrá algunas zonas privadas mediante el acceso con un usuario y clave que se pueden configurar.

Para configurar un usuario por primera vez se ejecutará la línea de comandos ***mosquitto_passwd -c /etc/mosquitto/passwd adminIoT,*** la ejecución de la línea de comandos se muestra en la figura 114.

```
root@debian10ServerIoT:/etc/mosquitto# mosquitto_passwd -c /etc/mosquitto/pwfile adminIoT
Password:
Reenter password:
root@debian10ServerIoT:/etc/mosquitto#
```

Figura 114. Creación de usuario en el Broker Mosquitto (Elaboración propia)

El comando mosquitto_passwd con la opción -c permite crear un archivo de contraseña con nombre passwd donde se almacena el usuario y la contraseña encriptada como se observa en la figura 115.

```
root@debian10ServerIoT:/etc/mosquitto# cd /etc/mosquitto/
root@debian10ServerIoT:/etc/mosquitto# ls -l
total 20
drwxr-xr-x 2 root root 4096 Nov 12 23:26 ca_certificates
drwxr-xr-x 2 root root 4096 Nov 12 23:28 certs
drwxr-xr-x 2 root root 4096 Nov 15 10:37 conf.d
-rw-r--r-- 1 root root  348 Aug 19 09:03 mosquitto.conf
-rw-r--r-- 1 root root  118 Nov 15 10:44 pwfile
root@debian10ServerIoT:/etc/mosquitto# cat pwfile
adminIoT:$6$zTjvnsTp7GZbqGsu$DjelCWpP1ZMn0tiZeSAupP0Nq/yyFjQwiI3DZRlzNHzbvGa5qeXG9pJ/y49KkTxzMhs4sr93zflWWdjwDBlICw==
root@debian10ServerIoT:/etc/mosquitto#
```

Figura 115. Archivo pwfile creado y contenido del archivo (Elaboración propia)

En la figura 115 se muestra el archivo pwfile que se encuentra en el directorio /etc/mosquitto/, dentro del archivo se muestra el usuario creado como adminIoT y la clave que se encuentra en un formato de encriptación por seguridad.

Para la creación de un usuario adicional y almacenarlo en el archivo se ejecuta la línea de comandos ***mosquitto_passwd /etc/mosquitto/pwfile User-Uno***. A diferencia de la línea de comandos anterior no se selecciona la opción -c que es la que realiza la creación de un nuevo archivo de contraseña. Como se puede ver en la figura 116, se muestra la ejecución

de la línea de comandos y en el contenido del archivo de contraseña ya aparece el usuario adicional añadido.

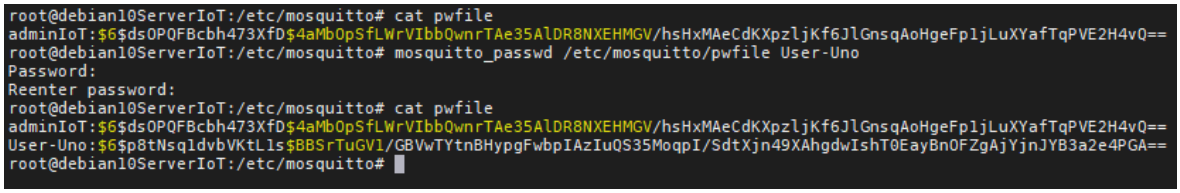

Figura 116. Usuario adicional creado en el archivo de contraseña (Elaboración propia).

Para configurar la autenticación desde el archivo creado pwfile, se debe cargar un archivo de configuración ***mosquitto.conf***, en el directorio ***/etc/mosquitto/config.d/***. El archivo de configuración se encuentra comprimido como mosquitto.conf.gz dentro del directorio ***/usr/share/doc/mosquitto/examples/***, se descomprime el archivo y se copia al directorio antes mencionado para poder editarlo con el editor de texto nano. Este archivo se muestra dentro del directorio /etc/mosquitto/conf.d/ en la figura 117.

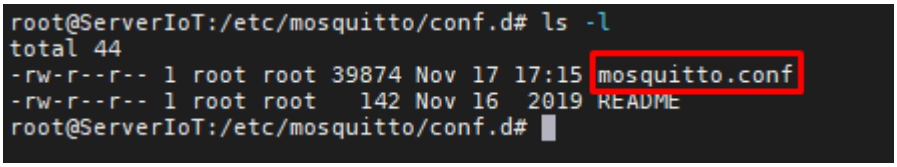

Figura 117. Archivo mosquitto.conf copiado (Elaboración propia).

Una vez copiado el archivo se procede a editar para realizar la configuración, se busca dentro del archivo la sección Security y al final de esta sección se añade las líneas:

password_file /etc/mosquitto/passwd
allow_anonymous false

La primera línea indica el directorio donde se buscará la información de usuario y contraseña, y la segunda línea deshabilita todas las conexiones no autenticadas. La configuración del archivo se muestra en la figura 118.

```
# =================================================================
# Security
# =================================================================

# If set, only clients that have a matching prefix on their
# clientid will be allowed to connect to the broker. By default,
# all clients may connect.
# For example, setting "secure-" here would mean a client "secure-
# client" could connect but another with clientid "mqtt" couldn't.
#clientid_prefixes

# Boolean value that determines whether clients that connect
# without providing a username are allowed to connect. If set to
# false then a password file should be created (see the
# password_file option) to control authenticated client access.
#
# Defaults to true if no other security options are set. If any other
# authentication options are set, then allow_anonymous defaults to false.
#
#allow_anonymous true
allow_anonymous false
password_file /etc/mosquitto/passwd

# -----------------------------------------------------------------
# Default authentication and topic access control
# -----------------------------------------------------------------
```

Figura 118. Configuración de archivo mosquitto.conf (Elaboración propia)

3.3. Diseño y construcción de una interfaz gráfica que permita la interacción con la estación remota y el usuario para el monitoreo y control de los equipos de la estación repetidora.

Para el monitoreo y control de la estacion remota se debe tener un interfaz que permita mostrar los niveles de temperatura, potencia del transmisor o transmisores en el caso que se disponga de uno de respaldo, estado de la red eléctrica y adicional realizar un encendido o apagado de la estacion mediante el control de un contactor que controla la alimentación hacia el transmisor.

Para configurar esta interfaz, se va a utilizar una aplicación que trabaja con el sistema operativo Android que se llama ***Linear MQTT Dashboard*** mostrada en la figura 119, esta aplicación es un cliente MQTT en la que se puede configurar una interfaz gráfica denominada Dashboard (tablero), que permite integrar widgets[2] como interruptores, botones, leds indicadores, deslizadores, títulos, metros de lecturas, gráficos en función del

[2] Widget, pequeña aplicación que se ejecuta por un motor de widgets, su función es dar fácil acceso a funciones que se usas frecuentemente, así como también de proveer información visual.

tiempo y conjuntos de botones para control. Cada uno de estos widgets que se incorporan al tablero deben suscribirse a un topic (tema) para que permitan mostrar información del cliente que publique en un determinado topic.

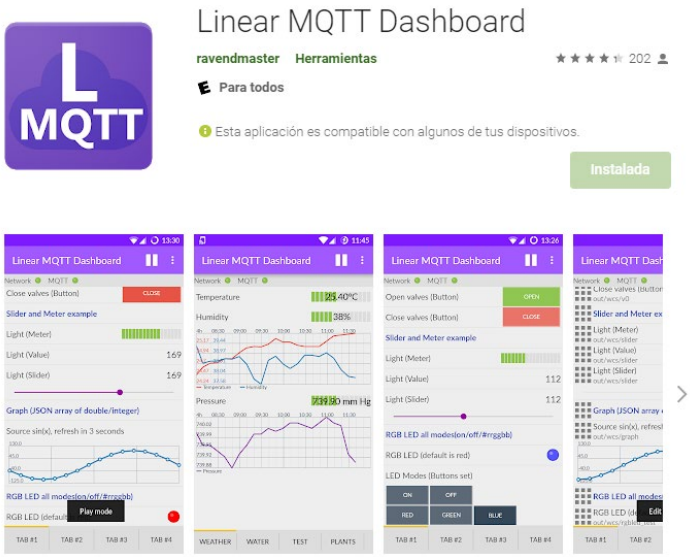

Figura 119. Aplicación Linear MQTT Dashboard *(PlayGoogleStore, Linear MQTT Dashboard, 2020)*

Esta aplicación también permite la publicación de mensajes en el bróker para ser leídos por otros clientes que se suscriban al topic correspondiente, esta interacción se lo hace fácilmente debido a que la aplicación tiene un interfaz amigable e intuitiva y de fácil configuración. De esta manera se puede obtener información de los parámetros de la estacion que se encuentra monitoreando y además controlar ciertos equipos que se puedan conectar a los actuadores para hacer un reinicio del sistema o reinicio de equipos independientes, dependiendo de la conexión realizada en la estacion. En la figura 120 se observa los widgets disponibles dentro de esta aplicación.

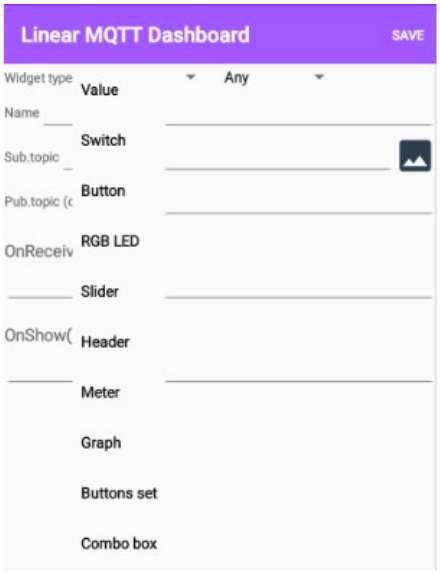

Figura 120. Widgets posibles a seleccionar (Elaboración propia).

La aplicación mostrada en la figura 119, tiene un interfaz amigable que permite la configuración del tablero de manera intuitiva, como se puede ver en la figura 120 se puede integrar varios widgets que pueden cumplir varias funciones dependiendo el tipo que se escoja, estos widgets permiten la suscripción o publicación en un topic determinado para mostrar la información de los sensores o enviar un mensaje para activar o desactivar un sistema mediante un actuador.

Una de las desventajas que se tiene en la aplicación es que únicamente permite conectarse a un bróker, en el caso de requerir conectarse a otro bróker diferente no se permite; sin embargo, existe otra aplicación que si permite la conexión a diferentes bróker, la aplicación trabaja con el Sistema operativo Android, su nombre es ***Nonlinear MQTT Dashboard*** que se muestra en la figura 119, en este caso del proyecto presentado; no es necesario conectarse a un bróker diferente debido a que se puede ampliar reestructurando la creación de topics para diferentes tipos de estaciones y así trabajar únicamente con un solo bróker.

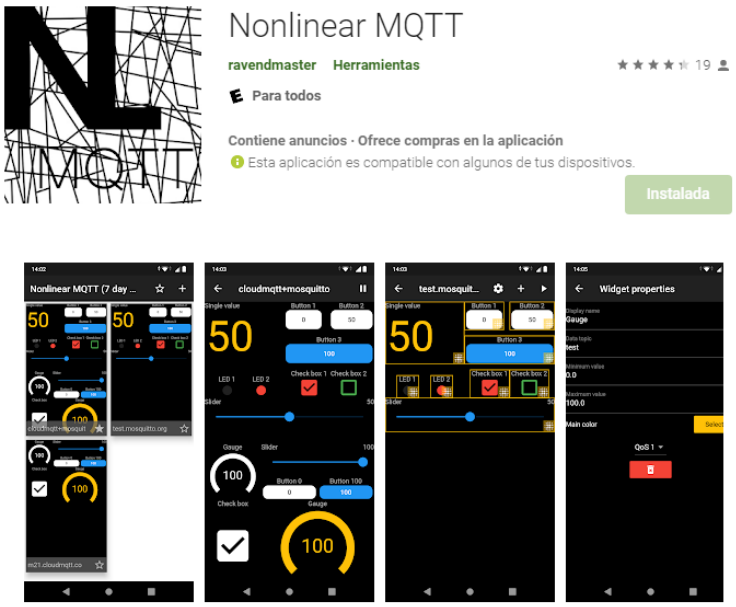

Figura 121. Aplicación para Android Nonlinear MQTT *(PlayGoogleStore, , 2020)*.

3.3.1 Configuración del Tablero en Linear MQTT Dashboard

Para el diseño del tablero se va a colocar un título en la sección principal que identificara la estacion o el sitio que se está monitoreando, mediante el widget Header que permite escribir un título como se muestra en la figura 120.

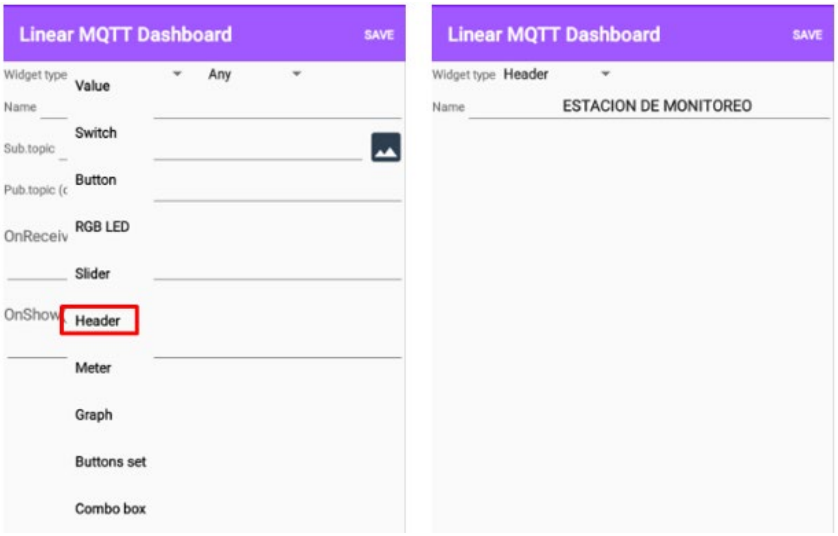

Figura 122. Widget Header para identificar con títulos (Elaboración propia).

Luego se creará secciones que identificará cada sensor, se distribuirá de la siguiente manera: Temperatura y Humedad actual, Estado transmisor principal, Estado transmisor respaldo, Estado red eléctrica y Reinicio del sistema.

Para la primera sección de temperatura y humedad actual se colocará un título con este nombre, se usará el widget Header para insertar el título como se muestra en la figura 121.

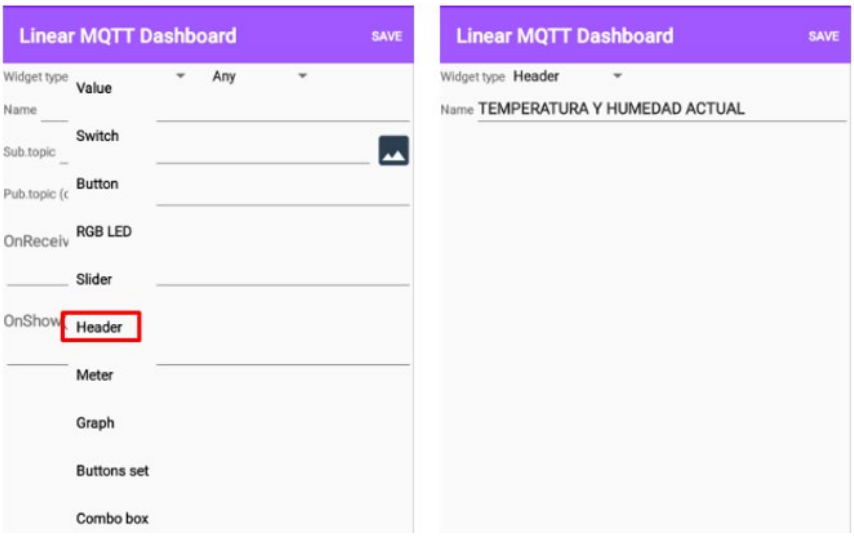

Figura 123. Widget Header para identificar sección (Elaboración propia).

En la segunda sección se configura dos widgets con Value, que muestren la información de la temperatura y humedad en forma visual, los valores de entrada se obtendrán de los topics /proyectoIoT/estacion/temperatura y /proyectoIoT/estacion/humedad como se muestran en la figura 122 y figura 123 respectivamente, también se configurar un nombre para identificar la medida del sensor que se está conectando.

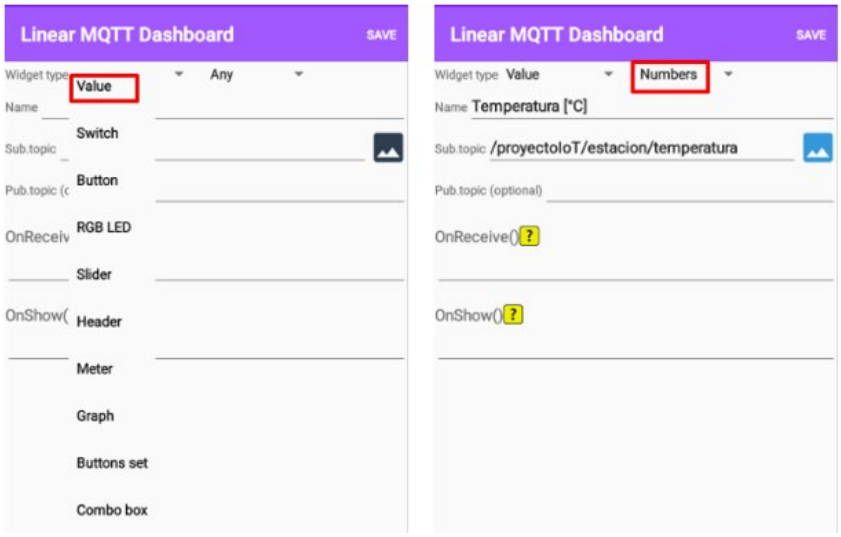

Figura 124. Widget Value para mostrar datos de temperatura (Elaboración propia).

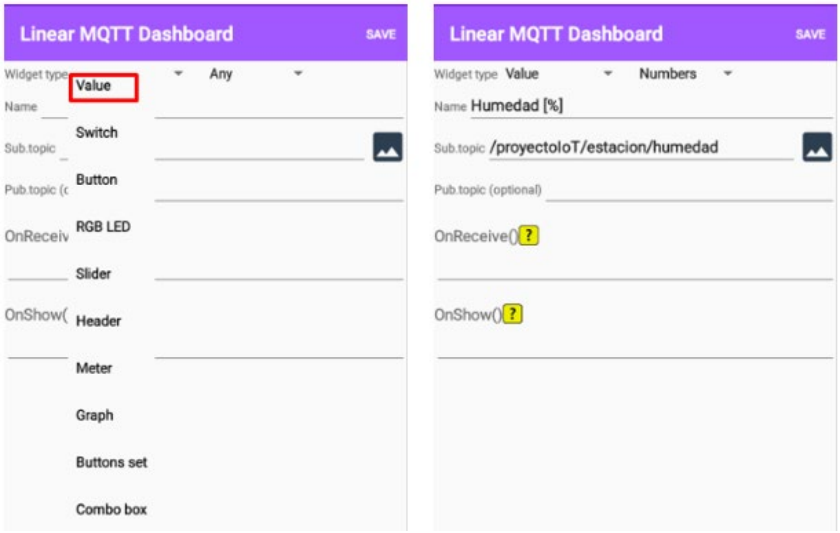

Figura 125. Widget Value para mostrar datos de humedad (Elaboración propia).

En la segunda sección de Estado Transmisor Principal se insertará un título con este nombre con el widget Header como se muestra en la figura 124.

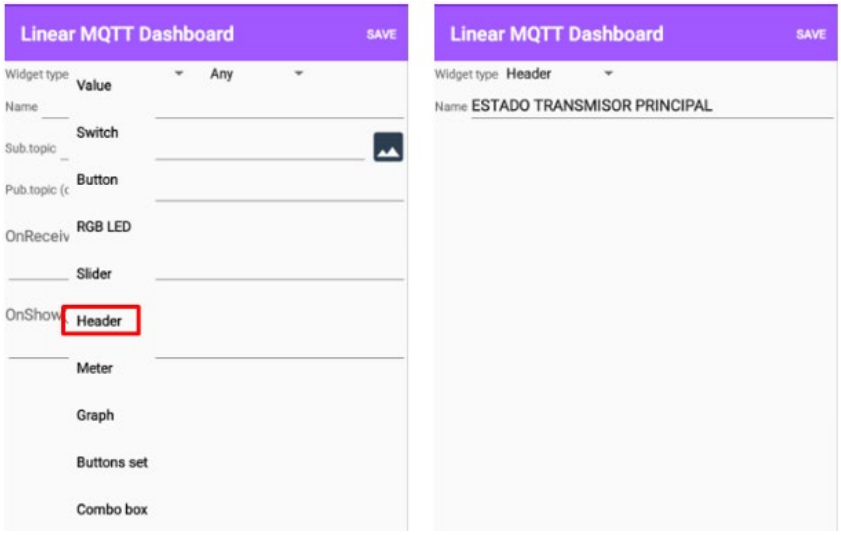

Figura 126. Widget Header para identificar la sección (Elaboración propia).

A esta sección también se añadirá un widget de Value, para mostrar el valor numérico de la potencia a la que se encuentra operando el transmisor y de igual manera se debe suscribir a un topic, que para este widget será /proyectoIoT/estacion/transmisorUno como se muestra en la figura 125. Adicional se insertará un título que identifique la medida que se está adquiriendo.

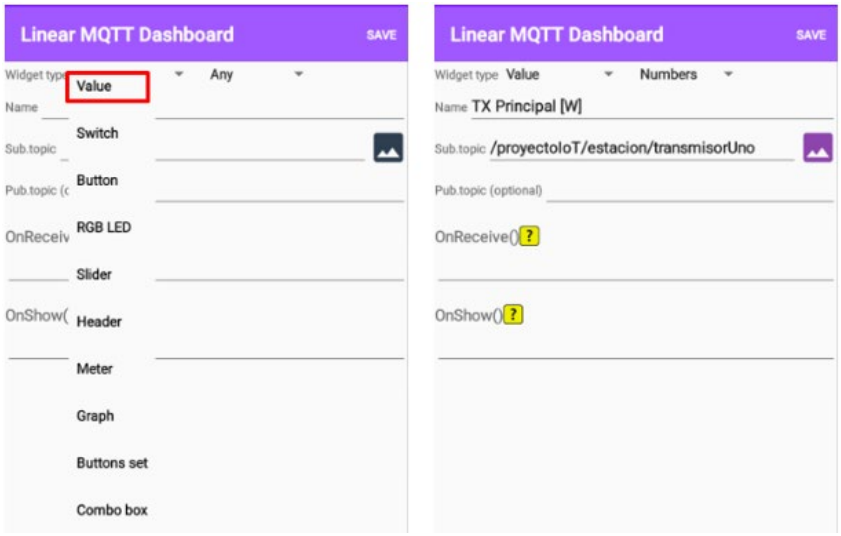

Figura 127. Widget Value para mostrar Potencia Tx Principal (Elaboración propia).

También se añadirá un widget Switch, que enviara un mensaje para habilitar o deshabilitar el transmisor principal, el topic en el que se publicara en mensaje es

/proyectoIoT/estacion/activarUno, con esto se logra un control sobre el transmisor principal como se muestra en la figura 126.

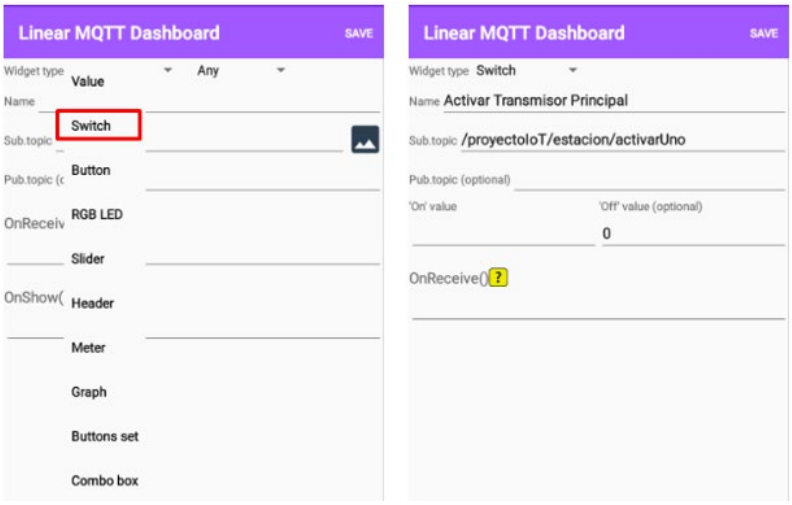

Figura 128. Widget Switch encender y apagar el transmisor (Elaboración propia).

La sección tres es igual que la sección dos, se añade un widget Header, Value y Switch. Con Header se inserta un título con el nombre Estado Transmisor Respaldo para identificar la sección, con Value se muestra el nivel de potencia del transmisor de respaldo y se suscribe al Topic /proyectoIoT/estacion/transmisorDos para obtener la información de medida que se recibe desde el transmisor de respaldo. Con el widget switch se puede enviar un mensaje para activar o desactivar un actuador que a la vez se conecta a un equipo o sistema. Toda esta configuración se muestra en las figuras 127, 128 y 129 respectivamente.

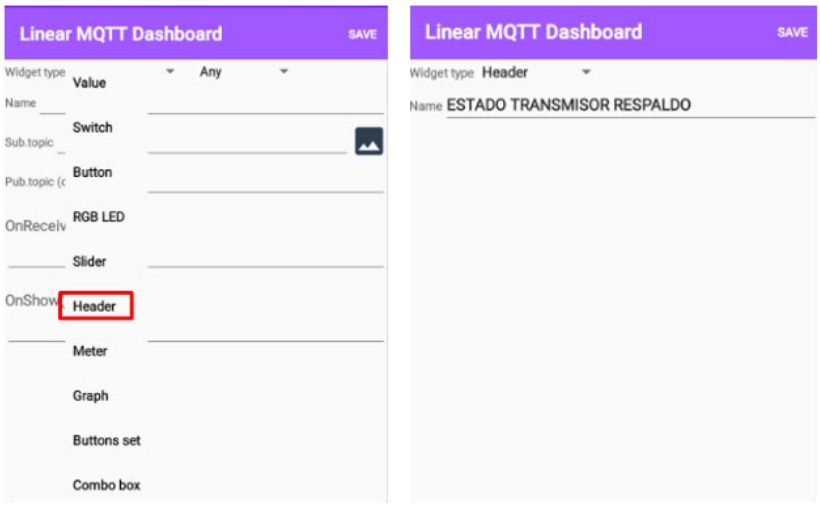

Figura 129. Widget Header para identificar la sección (Elaboración propia).

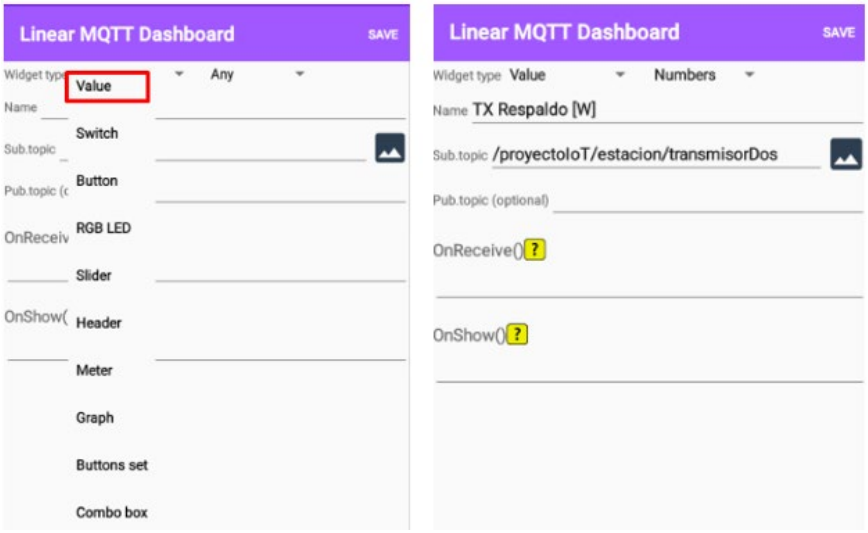

Figura 130. Widget Value para mostrar nivel de Potencia de TX de respaldo (Elaboración propia).

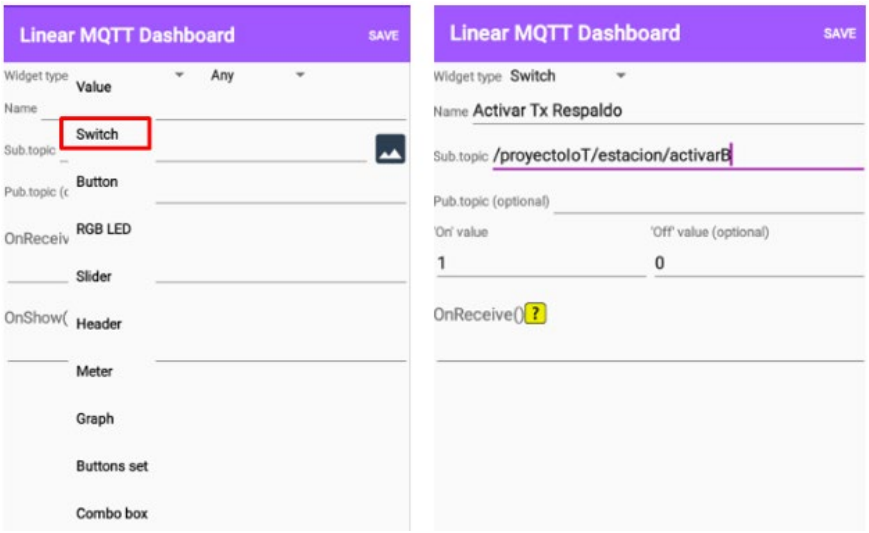

Figura 131. Widget Switch para activar y desactivar Tx de respaldo (Elaboración propia).

En la sección cuatro Estado de red eléctrica se insertará un título con este nombre para identificar la función que se realizara, se usara el widget Header para insertar el título como se muestra en la figura 130, adicional se usara el widget Value para mostrar el mensaje que se enviara por el topic /proyectoIoT/estacion/redelectrica como se muestra en la figura 131, este mensaje hará referencia al estado de la red eléctrica en la estacion, indicara con

un FAIL si alguna de las fases o a la vez las dos fases estén fuera de operación por algún problema y con un OK si el sistema eléctrico está operando correctamente.

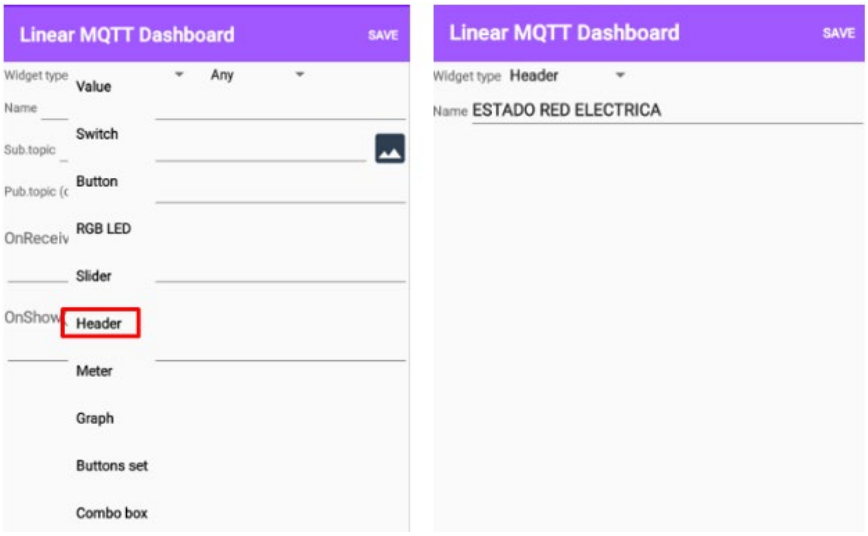

Figura 132. Widget Header para identificar la sección (Elaboración propia).

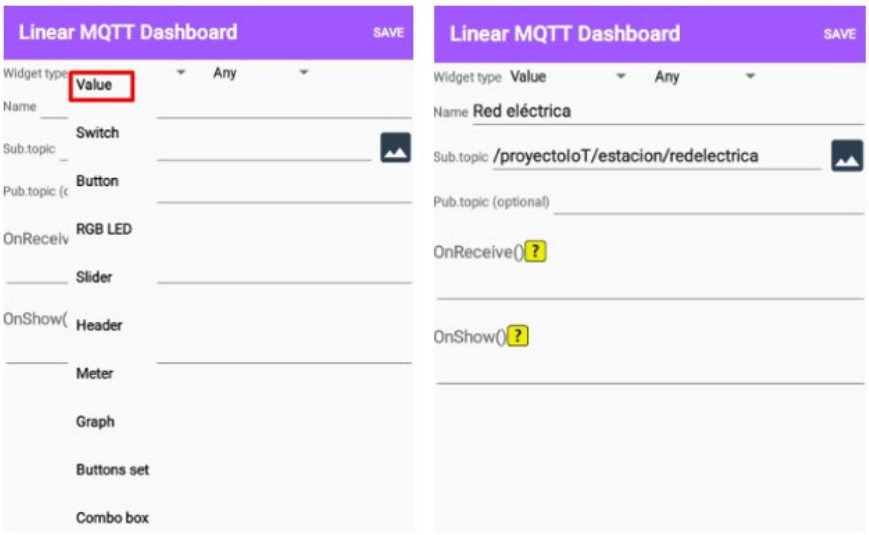

Figura 133. Widget Value para mostrar el estado de red eléctrica (Elaboración propia).

Finalmente, se crea una quinta sección que permitirá una interacción directa con el sistema eléctrico, esta sección controlara un contactor, que es el encargado de alimentar a todo el sistema de transmisión o a su vez al transmisor completo que se va a controlar. Para esto se utilizará el widget Header para insertar un título que identifique la sección como se muestra en la figura 132, adicional se usará el widget Button, que enviará un mensaje de

reinicio mediante la publicación al topic /proyectoIoT/estacion/reinicioEquipo como se muestra en la figura 133.

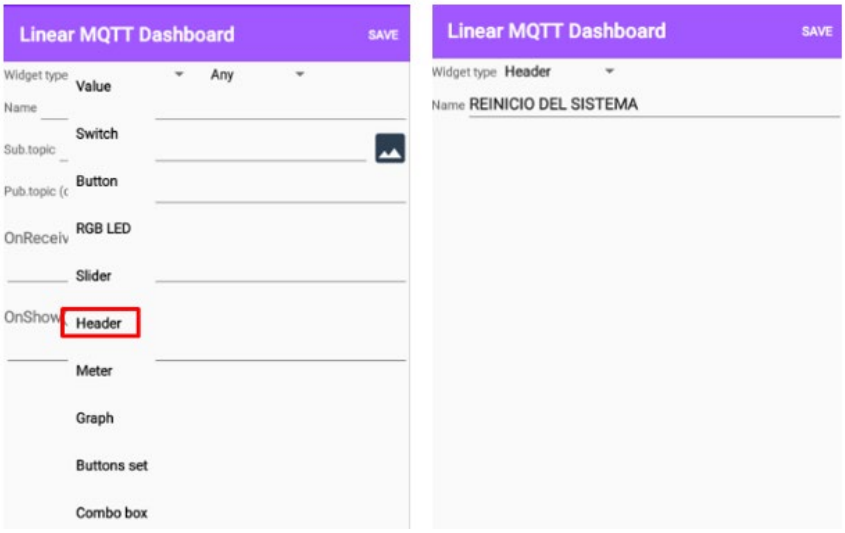

Figura 134. Widget Header para identificar la sección (Elaboración propia).

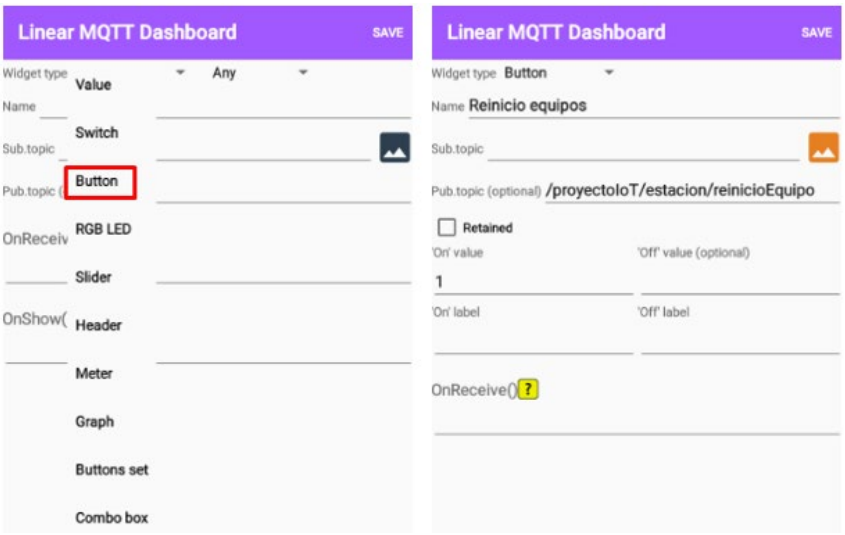

Figura 135. Widget Button para publicar un mensaje de reinicio (Elaboración propia).

Toda la configuración realizada anteriormente es guardada automáticamente, y queda disponible en la aplicación cuando se requiera revisar parámetros de la estacion que se monitorea, o realizar alguna conmutación en los equipos de transmisión entre el principal y el de respaldo. La configuracion final del tablero se muestra en la figura 134, ahí se observa las secciones configuradas anteriormente, con cada titulo correspondiente para identificar el parametro que se esta midiendo.

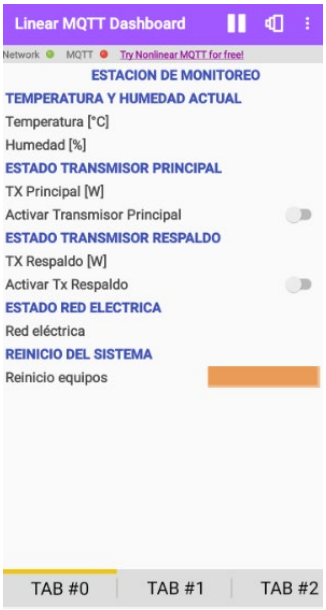

Figura 136. Aplicación funcionando con los widgets configurados (Elaboración propia).

3.4. Diseño y construcción de un sistema de comunicaciones que permita la transmisión de datos desde la estación remota hacia el usuario.

Para la comunicación de los sensores conectados al módulo Arduino y el Broker MQTT creado en la nube, se va a utilizar el módulo de Arduino Shield Ethernet W5100 que se muestra en la figura 135, este módulo tiene una conexión RJ-45 estándar que permite conectarse a la red de forma cableada asegurando una conexión estable. Una vez conectado en la red y si está disponible la conexión a internet este comienza a transmitir datos hacia fuera donde se encuentra configurado el servidor. El bróker comienza a recibir esta información mediante el puerto 1883.

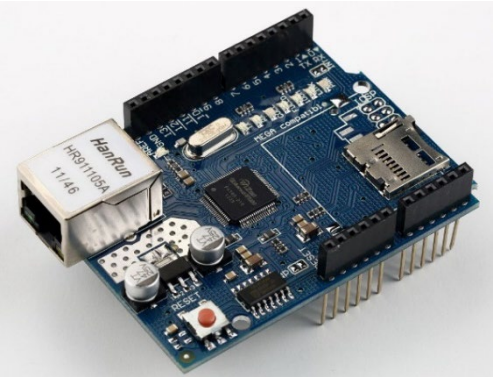

Figura 137. Modulo Shield Ethernet Arduino W5100 *(Ahedo Mardones & Ahedo Gonzáles, 2019)*.

Para utilizar el módulo se requiere instalar la librería **Ethernet** en Arduino IDE que se muestra en la figura 136, esta librería permite conectarse a internet mediante la configuración en el código de programación, provee funcionalidades de cliente y servidor. Se puede configurar dentro de una red para adquirir dirección IP dinámica mediante DHCP o configurar una dirección IP estática, para el caso de la configuración a realizar se requiere una dirección IP estática, para asegurar una conexión sin problemas.

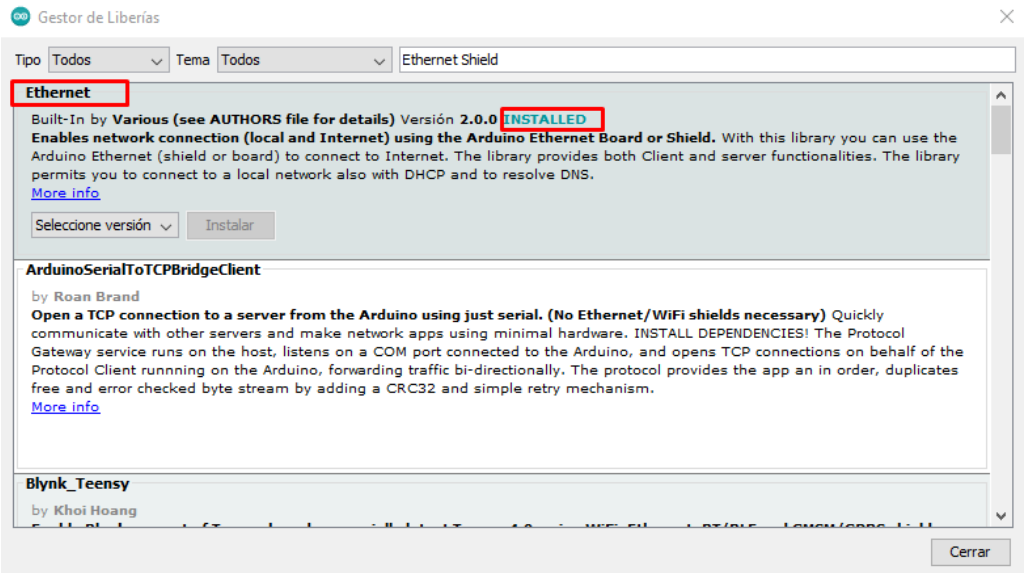

Figura 138. Librería Ethernet instalada (Elaboración propia).

Las librerías que se debe habilitar dentro del código de programación es la de <Ethernet.h>, una vez habilitado esta librería se debe configurar una MAC para el Arduino y una dirección IP estática que se encuentre dentro del rango de direcciones IP de la red que va a proporcionar salida hacia el Internet. Adicional, se configura la dirección IP pública que proporciona el servidor en la nube para poder conectarse a este mediante las instrucciones

que se escriba en el código de programación, la IP del servidor que se configura permitirá trabajar con la librería <PubSubClient.h> que permitirá la publicación y suscripción de un cliente en el bróker MQTT. Luego se procede con la declaración de variable para el cliente ethernet, luego se inicializa el módulo mediante la instrucción Ethernet.begin(mac, ip), en esta instrucción se carga la dirección MAC e IP configurados anteriormente y así queda inicializado el módulo y operativo, lo mencionado se puede mostrar en la figura 137.

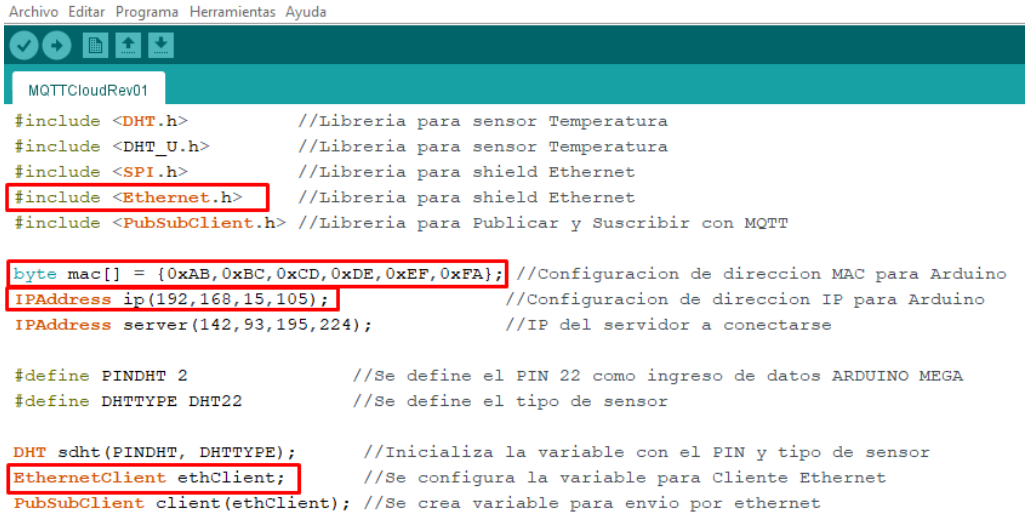

Figura 139. Configuración de tarjeta Shield Ethernet (Elaboración propia).

Para la conexión a internet que se deberá tener en la estacion repetidora podrá ser mediante un enlace de datos, para el proyecto que se presenta se utilizará una red de datos que corresponda a cualquier operador de telefonía móvil que existe en la actualidad en el país.

Para esta conexión hacia la red de datos se requiere de un módulo que realice esta funcionalidad, por lo que se decidió utilizar un router que permita usar una red móvil y para que esto suceda debe tener un socket para la inserción de una tarjeta SIM de cualquier operadora.

A causa de lo mencionado anteriormente se puede encontrar varios routers en diferentes marcas y modelos que cumplen la función de conectarse a una red móvil, se escogió un router con las características descritas en la tabla 10, debido a que lo único que se requiere para el proyecto, es que se pueda conectar a la red celular y que proporcione salida hacia el internet para permitir la transmisión de datos obtenidos por los sensores.

4G N300 LTE Router	
Marca	D-Link
Modelo	DWR-M921
Serie	U401104000964

Tabla 10. Datos router 4G LTE N300

Las características adicionales se describen a continuación, así como la hoja de datos que se puede encontrar en el anexo 4.

- Integra una WAN ETH y 3G/4G-LTE (donde esté disponible) para conexiones de alta velocidad a Internet (D-Link, 2018).
- Con un puerto WAN Fast Ethernet conéctate a alta velocidad a Internet Fija Banda Ancha (D-Link, 2018).
- Tiene Cuatro puertos LAN Fast ETH para conectar dispositivos alámbricos de alta velocidad (D-Link, 2018).
- Soporta WiFi N (802.11bgn) para conectar en forma inalámbrica a PCs y dispositivos móviles (D-Link, 2018).
- Su puerto USB 2.0 permite compartir Multimedia de un dispositivo de almacenamiento (Dispositivo no incluido) (D-Link, 2018).
- Incluye Wi-Fi Protected Setup (WPS) para agregar dispositivos de manera rápida y segura en su Red Wireless (D-Link, 2018).
- WPA/WPA2 para tráfico encriptado seguro (D-Link, 2018).

En la figura 138 se muestra el router que se utilizara, así como también en la figura 139 se muestra la ubicación física donde se debe insertar la tarjeta SIM del operador que se vaya a utilizar. Una de las ventajas de este equipo es que viene homologado y abierto para cualquier operador que esté disponible dentro del país.

Figura 140. Router D-Link *(D-Link, 2018)*.

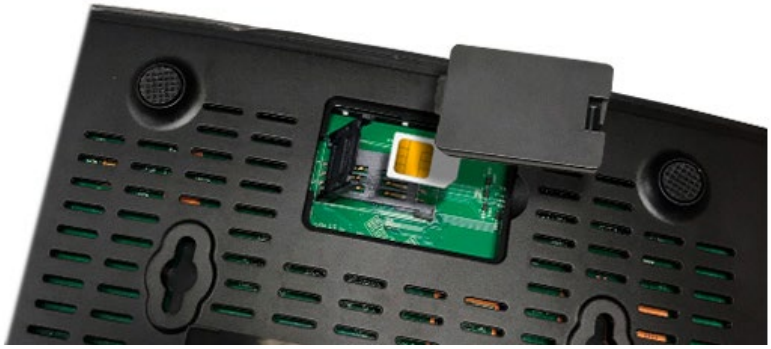

Figura 141. Ubicación de chip celular en el router *(D-Link, 2018)*.

3.5. Implementación y pruebas del sistema de monitoreo IoT

Para realizar la implementación del prototipo, se procederá a listar los materiales necesarios en la tabla, para poder llevar a la práctica todo lo mencionado anteriormente.

Ítem	Cantidad	Dispositivo	Descripción
1	2	Tarjetas Arduino Uno	Versión R3
2	2	Tarjetas Shield Ethernet	Ethernet para Arduino W5100
3	1	Tarjeta reguladora.	Regula voltaje entre 0 y 12VDC
4	2	Tarjetas de Relay	Módulos 2 Relay para Arduino
5	2	Baquelitas perforadas	Baquelitas de 7cm x 5cm
6	1	Fuente switching	12VDC – 5A / 110VAC – 220VAC
7	1	Caja metálica	20cm x 15cm x 10cm
8	1	Conector AC	Conector para panel
9	1	Interruptor AC	Interruptor con luz piloto.
10	1	Conector BNC	Para panel
11	1	Cable AC	Cable de computadora
12	1	Juego de cables	Para conexiones.
13	12	Borneras de conexión	3 pines
14	3	Borneras en ángulo	4 pines (hembra y macho)

Tabla 11. Materiales utilizados para la implementación del prototipo (Elaboración propia).

Se procede con la adecuación de la caja para instalar cada uno de los materiales antes mencionados, se hará orificios para insertar los módulos de la tarjeta Arduino y Shield Ethernet, para las borneras tipo ángulo, para un conector BNC de entrada para la señal de monitoreo de Radio Frecuencia, para el conector de AC y el interruptor con luz piloto. Las adecuaciones en la caja se pueden mostrar en la figura 142.

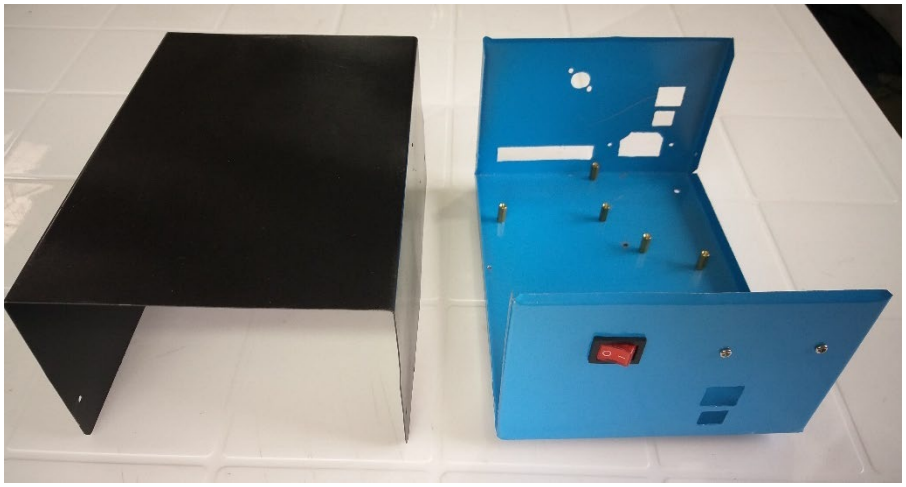

Figura 142. Adecuaciones realizadas a la caja (Elaboración propia).

Luego de realizadas las adecuaciones, se procede con la instalación del conector de AC de entrada, y la tarjeta con las borneras tipo ángulo para la conexión de los sensores y control de reinicio del sistema de transmisión, como se observa en la figura 143.

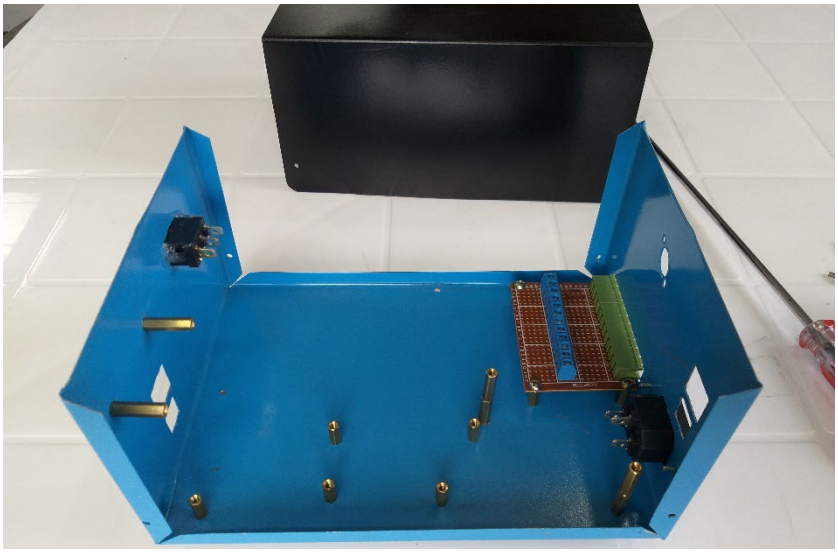

Figura 143. Instalación de interfaz de conexión sensores y control (Elaboración propia).

Se instala la fuente de poder de modelo TPS-1205 que soporta hasta 5A consumo conectado como carga en la fuente, seguidamente se instala la tarjeta reguladora de voltaje ajustable para que regule de 12VDC a 5VDC, esta tarjeta soporta consumos de hasta 3A, suficiente para operar sin problemas 2 tarjetas Arduino Uno y el sensor de temperatura. La instalación de estos módulos se muestra en la figura 144.

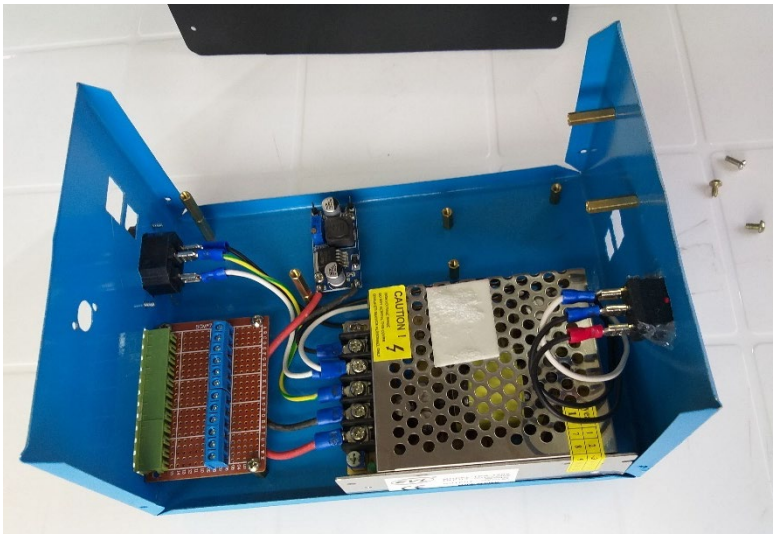

Figura 144. Instalación fuente de poder y regulador de voltaje (Elaboración propia).

Finalmente se instalan, los módulos de la tarjeta Arduino Uno y Shield Ethernet, que se conectaran a la red, un módulo de la Tarjeta Arduino se usara para la publicación de los mensajes en el broker MQTT y la otra tarjeta Arduino se usara para la suscripción de mensajes en el broker como se muestra en la figura 145.

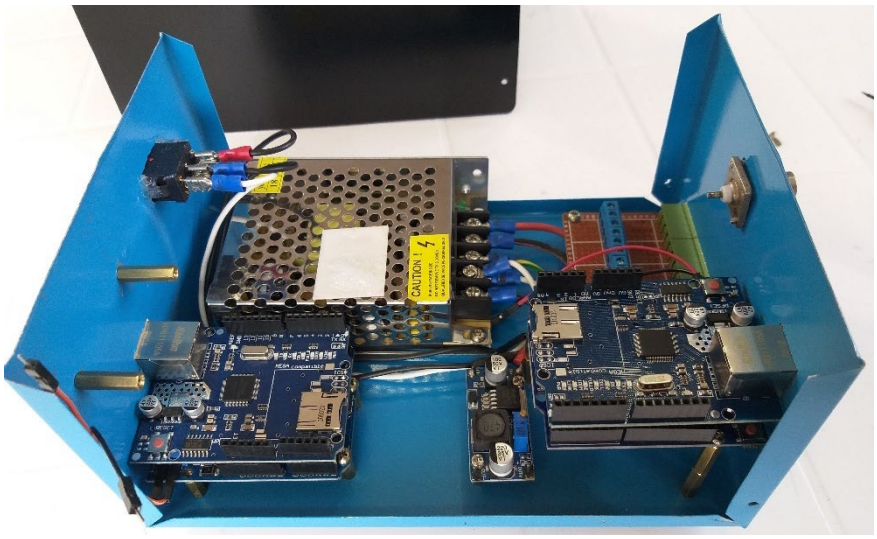

Figura 145. Instalación de tarjetas Arduino (Elaboración propia).

Adicional, se instalará dos tarjetas Relay de 2 canales cada una para controlar el reinicio del sistema y la activación de dos transmisores, principal y respaldo. Luego se realiza las conexiones de cada tarjeta y bloque instalados en la caja, de acorde con el diagrama de bloques presentado en la figura 146.

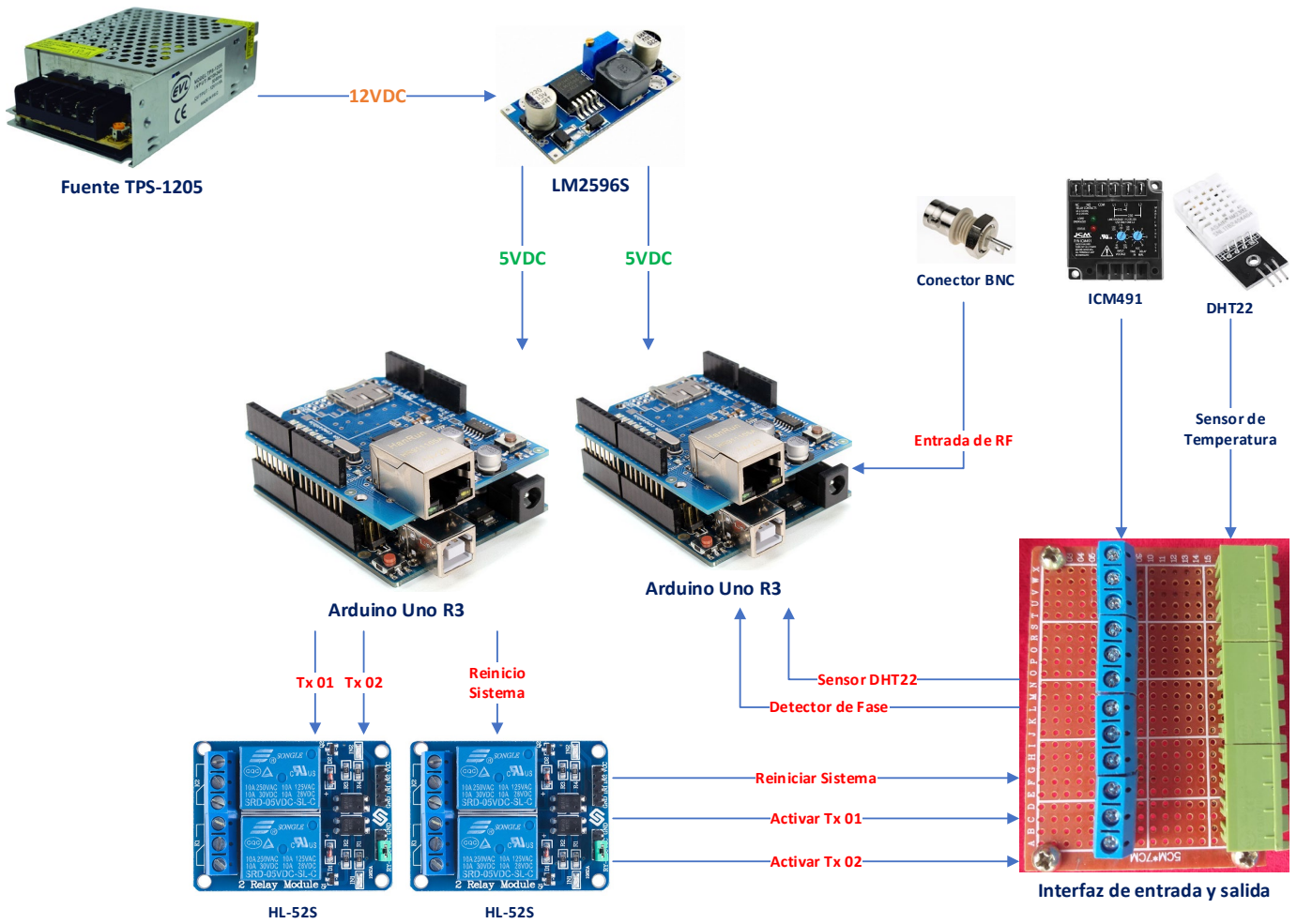

Figura 146. Conexionado de tarjetas en la caja (Elaboración propia).

Luego de realizar las respectivas conexiones en el interior de la caja como se observa en las imágenes de la figura 147, se encenderá toda la circuitería y se probará su funcionamiento.

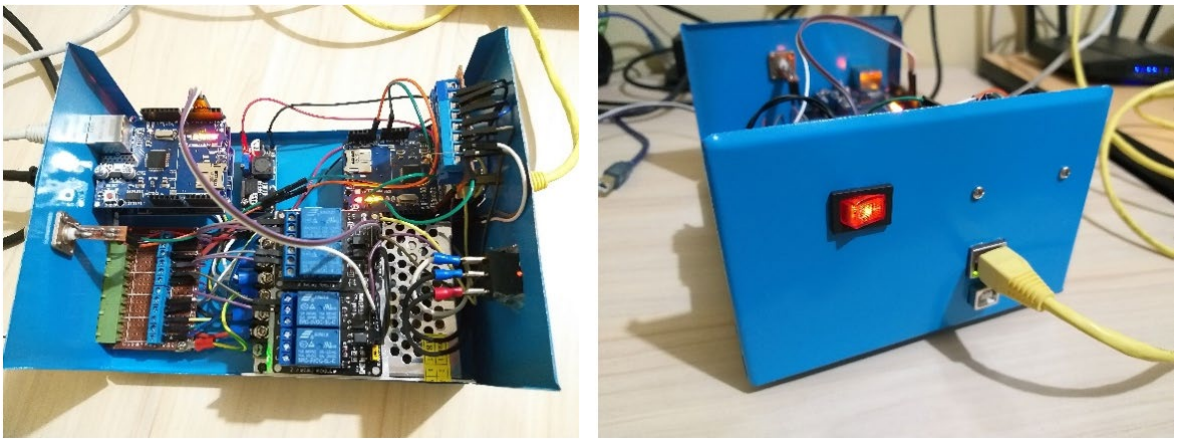

Figura 147. Pruebas de encendido de las tarjetas (Elaboración propia).

Cabe recalcar que para cada tarjeta de Arduino se cargan diferentes códigos de programación, en la primera tarjeta Arduino se cargara el código de programación para la publicación de mensajes en el broker, la publicación se hará cada 30 segundos; cada publicación que se realice se almacenara en la base de datos, esto quiere decir que existirá datos almacenados cada 30 segundos en cada una de las tablas de publicación en la base de datos. En la segunda tarjeta Arduino se cargará el código de programación para la suscripción a topics, que permitirá recibir los mensajes enviados por parte de los sensores.

Los códigos de programación que se cargaran en las tarjetas se encuentran en el ANEXO 5, ahí se muestra las funciones de programación ya descritas anteriormente.

Una vez cargado los códigos de programación, se realiza pruebas de funcionamiento del prototipo, para verificar el funcionamiento mediante la aplicación configurada en una sección anterior, se procede a suscribirlo con clave y usuario configurado para poder acceder a la información de la publicación de los topics respectivo como se observa en la figura 148.

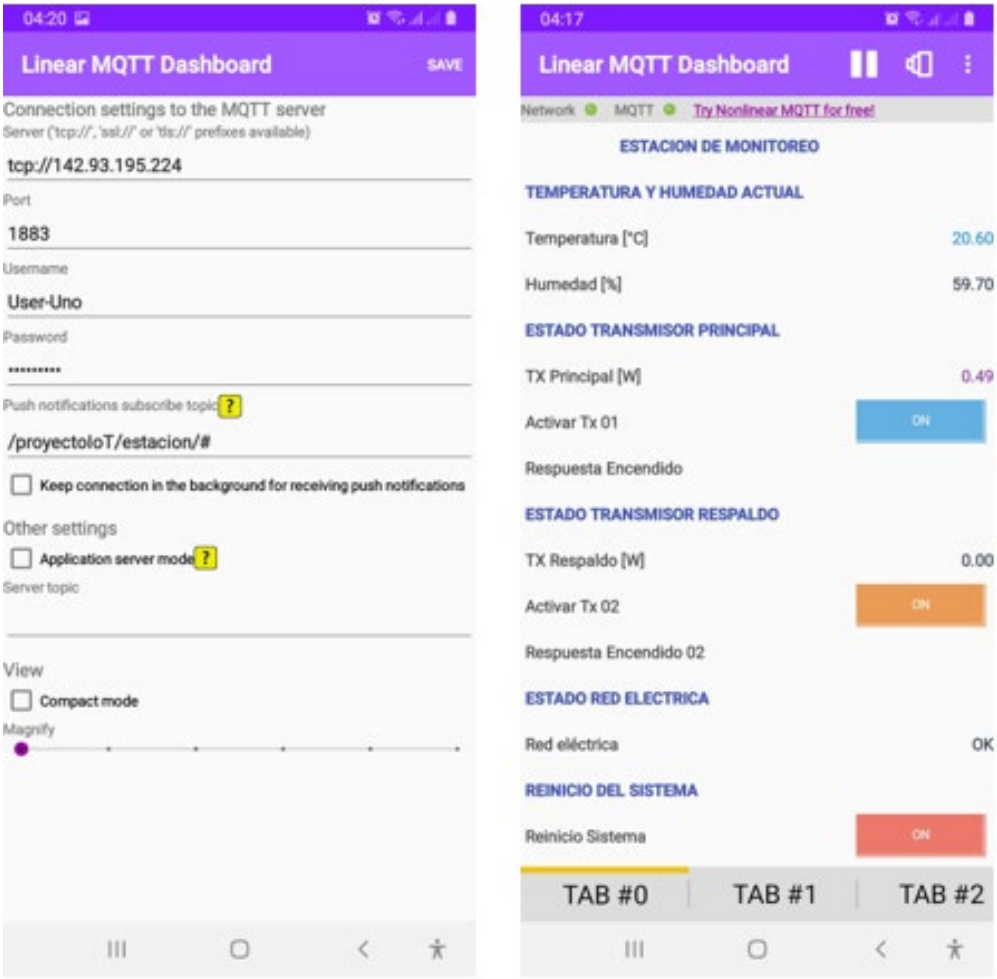

Figura 148. Suscripción al topic mediante autenticación (Elaboración propia).

Adicional, se muestran la ejecución de los widgets de botón para reiniciar el sistema, y para activar y desactiva el transmisor principal y respaldo, el funcionamiento se muestra en las imágenes de la figura 149. En las imágenes de la figura 149 se muestra la temperatura y humedad, valores tomados por el sensor DHT22, también se muestran los valores de potencia de transmisor principal y respaldo, en la imagen de la derecha se muestra la activación de los transmisores como Tx01-ON y Tx02-ON y en la imagen de la izquierda se muestra el apagado de los transmisores como Tx01-OFF y Tx02-OFF y finalmente se muestra el estado de la fase en la red eléctrica y muestra que es correcto.

En la figura 150 se muestra el prototipo funcionando sin problemas. También en la figura 151 se puede mostrar una captura de la base de datos “estaciondb” en la nube y los datos almacenados hasta el momento.

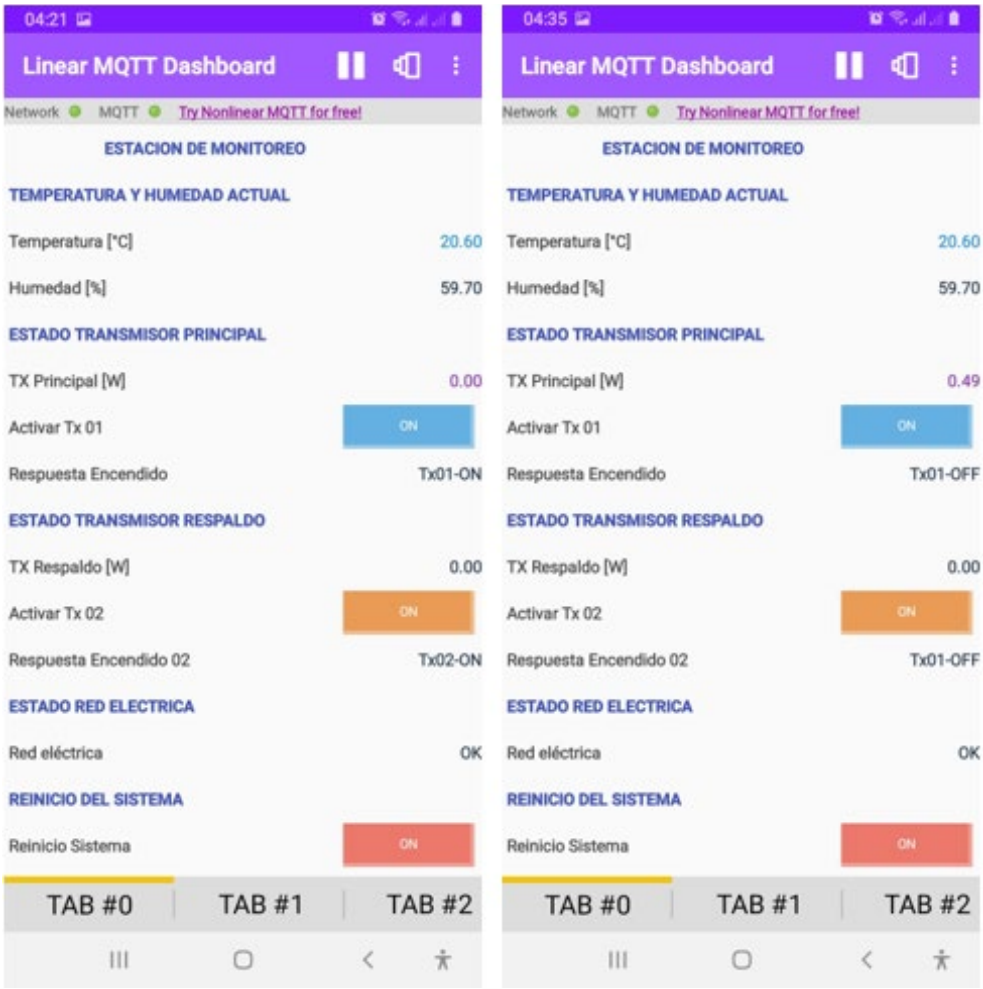

Figura 149. Funcionamiento de plataforma sistema IoT (Elaboración propia).

Figura 150. Prototipo operativo (Elaboración propia).

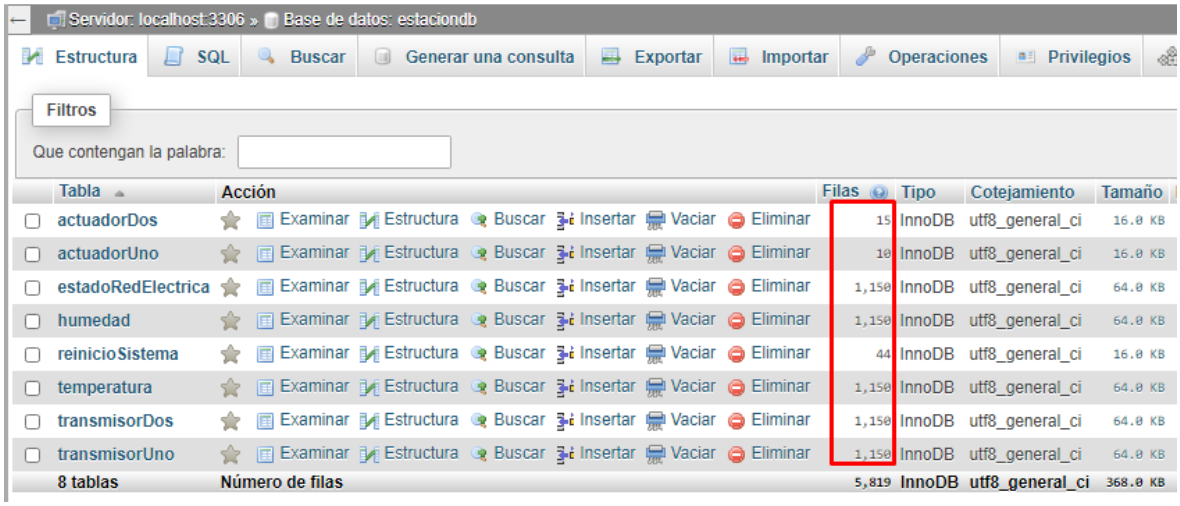

Figura 151. Datos almacenados en la base de datos (Elaboración propia).

CAPÍTULO IV

4. CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

- Se puede concluir que, de acuerdo a los análisis de trabajos previos realizados, el Internet de las cosas es implementado en proyectos de diferentes campos que permiten la recolección de información mediante sensores para no tener que estar físicamente en un lugar e inclusive se puede activar o desactivar un sistema mediante control remoto.
- Se puede concluir que el Internet de las cosas se puede aplicar al campo de las Telecomunicaciones, en el área de monitoreo y control de los sistemas de transmisión en sitios remotos, para poder optimizar tiempos de respuesta a fallas que se puedan solucionar sin tener que estar presente en el sitio.
- Un sistema de monitoreo y control de este tipo puede llegar a ser de utilidad, para evitar sanciones por parte del ente regulatorio, cuando exista fallas en una estacion y no se dé pronta solución. Tomando en cuenta que pudo solucionarse con un simple reinicio del sistema.
- El Internet de las cosas es una tecnología que hoy en día es bastante útil en la automatización de sistemas, donde cada cosa se puede conectar a una red, ya sea esta interna o externa, para poder monitorear o controlar de manera remota mediante un dispositivo móvil. Y así poder conocer el estado del objeto o casa que se conecte en la red.
- La tecnología 5G será la base para conectar cada cosa hacia la red de Internet y poder controlar cualquier dispositivo desde cualquier lugar, esta tecnología permite la conexión de una cantidad no definida de dispositivos, que pueden mantener servicios paralelos dentro de un ancho de banda móvil amplio, estos dispositivos podrán estar conectados de manera permanente.
- Se puede concluir que para implementar una plataforma de IoT en un sitio remoto, se debe garantizar la cobertura de señal celular de cualquier operador, con esto se garantiza que la conexión de los sensores y actuadores que se instalen en cualquier

sitio, puedan conectarse al broker mediante el internet. Adicional se puede utilizar un enlace inalámbrico de datos para llegar con Internet; sin embargo, esto sería posible si la estacion matriz y la estacion repetidora se encuentran en la misma ciudad.

- MQTT es un protocolo de transporte de mensajería de publicación y suscripción entre el servidor y el cliente, es liviano y permite una comunicación casi en tiempo real, su implementación es fácil y su configuración depende de los requerimientos de cada cliente que se conecte al broker. Puede ejecutarse sobre TCP/IP o sobre otros protocolos que proporcione conexiones bidireccionales sin perdidas. Estas características hacen que sea un protocolo muy usado actualmente en la implementación de plataformas del Internet de las cosas.
- Se puede concluir que un sistema de monitoreo aplicando la Tecnología IoT puede ser implementado con plataformas que ofrecen proveedores en la nube, estos ofrecen la gestión y administración del servidor IoT, estas plataformas son de pago que depende de la cantidad de dispositivos permitidos para interconectarlos dentro de la plataforma y de la interacción del número de clientes que pueden requerir la información de los dispositivos sensores o actuadores.
- Se puede concluir que para implementar sensores que se conecten a internet y posteriormente al broker, se debe hacer con tarjetas que permitan la programación mediante librerías que incluyan el protocolo MQTT y de ser posible la última versión para evitar problemas de conectividad a causa de las versiones que manejan cada dispositivo.
- En el mercado, existen varios dispositivos que configurados correctamente se pueden utilizar para configurar e implementar una plataforma IoT, aplicado a cualquier campo que se desee utilizar. Por lo general, existen proveedores que alquilan el broker MQTT ya configurado y mediante la creación de cuentas, se puede conectar dispositivos clientes para utilizarlos en un proyecto, aunque lo ideal sería configurar uno mismo el servidor donde se almacene el broker MQTT.
- Dentro de la plataforma IoT es indispensable el broker MQTT, este es un servidor que interactúa y conecta los clientes para mantener actualizado la información en un topic, a este se suscribe el usuario como cliente para obtener información enviada por los sensores que se utilizan dentro de una red.
- La plataforma de IoT se implementa en la nube, para esto se debe contratar un proveedor IaaS que ofrezca el alquiler de infraestructura como servicio, así se

puede configurar y administrar el servidor uno mismo, y solo se paga por lo que se usa dentro de lo contratado.

- El Broker MQTT debe estar operativo 24/7, lo que significa que, si el servidor donde se configura el Broker tiene problemas, toda la plataforma IoT deja de funcionar, a causa de este inconveniente se decide configurar un broker en la nube con un proveedor que permita administrar el servidor y solo proporcione la Infraestructura en la nube.
- Una de las características de una plataforma IoT es que cada cliente que se conecte al broker debe conectarse con una IP a la red, esto quiere decir que se requiere un módulo ethernet, tarjeta de red o modulo WiFi por cada sensor o actuador que se implemente dentro de la red. Por este motivo, para el presente proyecto se utilizará un módulo ethernet para los mensajes de Publicación y un módulo ethernet para los mensajes de suscripción. Con esto se garantizará un funcionamiento óptimo en la red.
- Se concluye que, para realizar una buena comunicación entre los topics de publicación y suscripción, se debe realizar un diseño óptimo de topics que permita integrar varios sitios de monitoreo y control, que permita la expansión futura de lugares que se puedan incluir dentro de la plataforma. El diseño debe permitir la independencia de los sitios para no tener inconvenientes cuando se requiera publicar o suscribir un dato en la red.
- Se puede concluir que, para realizar una conexión de los sensores hacia el internet, se requiere tener un enlace de datos hacia el cerro donde se va a ubicar el prototipo de monitoreo. También se lo puede hacer mediante la red celular siempre y cuando exista cobertura celular en el sitio, caso contrario no se puede realizar una conexión de los sensores. Una opción adicional se puede dar a través de la señal satelital, que es una señal que siempre se tendrá presente en los cerros porque la señal de programación para transmitir se baja de satélite mediante un receptor satelital; el problema radica en que se requiere un ancho de banda en satélite lo cual es costoso, y equipos que puedan realizar la demultiplexación de la señal en caso de enviar multiplexada y equipos que puedan desempaquetar la señal de internet. Por lo que se escoge la opción de conectarse con un router que pueda conectarse directamente a una WAN por enlace de datos o a una red móvil mediante una tarjeta SIM de cualquier operador, preferentemente un operador que tenga mayor cobertura celular en la estacion.

- MQTT permite encriptación en la comunicación, el propósito es utilizar Cifrado SSL es para brindar autenticación, encriptación e integridad, para lograrlo se requiere de una firma digital (certificado) y llaves públicas y privadas. La tarjeta Arduino Uno y Arduino Mega, no permite encriptación debido a su baja capacidad de computación baja, mientras que para el broker MQTT es insignificante la capacidad de computación y puede ejecutar sin problemas. Por este motivo no se usa cifrado SSL en el proyecto quedando una opción abierta para su implementación mediante tarjetas que si permitan el cifrado SSL.
- El broker configurado en el servidor permite varias conexiones de cliente, lo que permite una expansión para nuevos topics que pueden crearse para cada sitio donde se va a implementar la plataforma, cada cliente que se suscriba para obtener información del monitoreo puede conectarse mediante un usuario y contraseña que se añade en el archivo de contraseñas creado en el directorio /etc/mosquitto con el nombre pwfile.
- Para establecer una conexión con el broker MQTT usando encriptación se debe instalar la librería SSLClient que agrega funcionalidad TLS (Transport Layer Security) a cualquier clase de cliente. Esta librería no está disponible para ser usada con Arduino Uno y Arduino Mega, pero si puede ser usada con otros modelos de tarjetas de Arduino como ARDUINO DUE, ARDUINO NANO 33 IoT, ARDUINO CERO y otras más.
- El consumo de corriente es uno de los factores importantes que se debe considerar cuando se implementa un diseño en la práctica, ya que si se dispone de una cantidad considerable de sensores el consumo de corriente puede llegar a superar el valor máximo de corriente que puede soportar una fuente de poder, para el caso del presente proyecto no posee muchos sensores debido a la simplificación en el código de programación de las tarjetas Arduino Uno, el consumo que se tiene es de 350mA como máximo que puede llegar a consumir todo el circuito; mientras que el circuito regulador de voltaje puede soportar hasta 3A de consumo, dejando un amplio margen de tolerancia.
- Dentro de un sistema de sensores es necesario disponer de un router que permita la conexión inalámbrica de cada uno hacia la red donde se conectara con el broker para la transmisión de datos que serán publicados y estarán disponibles en los respectivos topics a los que se suscriban los diferentes clientes que requieran de la información.

- A lo largo de la historia el protocolo MQTT ha ido evolucionando hasta llegar a ser uno de los más utilizados hoy en día para la implementación de casas inteligentes, conexión de dispositivos con los que se pueden interactuar y controlar, conocer datos de temperatura, humedad, etc. Su uso dentro de estas plataformas se debe a su bajo consumo de ancho de banda, su fácil implementación, manejo en calidad de servicio y seguridad en la transmisión de datos.
- Los objetivos dentro del proyecto se cumplieron dentro de los parámetros correspondientes del Sistema de monitoreo planteado para el presente proyecto, permitiendo la construcción de un prototipo que puede ser instalado dentro de una estacion remota de radio o televisión, considerando las recomendaciones que se describe dentro de este capítulo para evitar problemas de funcionamiento.

4.2. Recomendaciones

- Para realizar la implementación de una Plataforma IoT, se recomienda disponer de una tarjeta de Red o Wifi por cada sensor que se conecte al Broker, esto quiere decir que si se tiene un sensor de temperatura por ejemplo; debe haber un interfaz de red que conecte a este sensor de manera inalámbrica o alámbrica para proporcionar la información en forma de datos al broker y este posteriormente realizara el procesamiento para publicar a cualquier cliente que se suscriba en el topic correspondiente.
- Para la implementación de una plataforma IoT se recomienda tener un servidor que trabaje 24/7 los 365 días del año, ya que en el servidor se instalara el broker quien será el que administre las publicaciones y suscripciones de los clientes dentro de la red. Adicional se debería tener un respaldo de este servidor en caso que exista problemas con el servidor principal.
- Se recomienda asignar direcciones IP estáticas a cada sensor que se configure, así como al servidor que se configure; si no se configura con IPs estáticas, cuando se reinicie algún elemento del sistema adquirirá una IP diferente a la que tenía anteriormente y el funcionamiento de la plataforma se verá afectada.
- Cuando se vaya a realizar un proyecto más robusto de configurar una plataforma IoT para cualquier aplicación, se recomienda utilizar tarjetas de programación que acepten encriptación con el protocolo TLS (Transport Layer Security), ya que

Arduino Uno y Arduino Mega por la escasa capacidad de memoria no permite la implementación. Se puede usar tarjetas como Arduino DUE, Arduino MKR FOX 1200, Arduino MKR WiFi 1010, Arduino Cero, etc.

- Cuando se vaya a realizar un proyecto de IoT se recomienda conectar cada sensor mediante una interfaz inalámbrica para evitar el múltiple cableado que pueda existir, es decir mediante una red Wifi que contenga un router de buenas características donde se pueda concentrar todas las conexiones inalámbricas dentro de la red.
- Se recomienda que los datos recogidos de los diferentes sensores puedan ser almacenados en una base de datos, para que estén disponibles para un análisis que permita acciones futuras correctivas para la optimización de recursos dentro de una organización.
- Cuando se vaya a implementar una plataforma de IoT para cualquier tipo de proyecto, se recomienda realizar un circuito independiente por cada sensor que se conecte como cliente, esto quiere decir su propia fuente de alimentación y conexión hacia la red interna. Esto permitirá que el sistema se vuelva escalable y se pueda aumentar la cantidad de sensores que sea necesario sin alterar a otros y adicional cuando uno de estos falle solo se cambiara el sistema del sensor dañado.
- Cuando se vaya a instalar un sensor o actuador dentro de la red se recomienda que se use la red WiFi, para evitar problemas de cableado que pueda ocasionar a futuro como cable en mal estado y se tenga que reemplazar, adicional evitar la contaminación de señales que se puedan inducir por el cable y afecten directamente a la tarjeta de programación.
- Para la instalación del prototipo se recomienda un pequeño UPS que pueda mantener la autonomía luego de un corte de luz, puede ser de 420/620 VA (Volta-Amperios), considerando que el prototipo consume menos de un amperio, por lo que la autonomía del UPS puede durar el tiempo suficiente para que indique que hubo un corte de energía eléctrica mediante el sensor de detección de energía eléctrica.

BIBLIOGRAFÍA

Ahedo Mardones, J. L., & Ahedo Gonzáles, A. (7 de abril de 2019). *Como funciona el módulo Arduino Ethernet Shield*. Obtenido de web-robotica: https://www.web-robotica.com/arduino/como-funciona-el-modulo-arduino-ethernet-shield

Alex. (10 de Marzo de 2019). *Cambiatealinux.com*. Obtenido de ln - crear un enlace simbólico al fichero o directorio: https://cambiatealinux.com/ln-crear-un-enlace-simbolico-al-fichero-o-directorio

Andrés, R. (27 de noviembre de 2018). *Usos de la Raspberry Pi que no sabías que podías darle*. Obtenido de Computer Hoy: https://computerhoy.com/noticias/hardware/15-usos-raspberry-pi-que-no-sabias-que-podias-darle-74905

Arduino. (1 de enero de 2020). *Arduino Mega 2560*. Obtenido de Arduino: https://store.arduino.cc/usa/mega-2560-r3

Arequipa Cunalata, D. A. (1 de julio de 2019). *Desarrollo de un prototipo de sistema de seguridad contra intrusos utilizando protocolos de IoT sobre la plataforma Zolertia Remote.* Obtenido de Biblioteca Digital EPN: https://bibdigital.epn.edu.ec/handle/15000/20318

Azzola, F. (18 de noviembre de 2018). *Protocolo CoAP: Guía paso a paso*. Obtenido de DZone: https://dzone.com/articles/coap-protocol-step-by-step-guide

Cárdenas, A. (28 de noviembre de 2016). *¿Qué es una plataforma IoT?* Obtenido de Secmotic: https://secmotic.com/plataforma-iot/

CloudMQTT. (16 de Febrero de 2020). *Cloud MQTT Documentacion*. Obtenido de Cloud MQTT: https://www.cloudmqtt.com/

ComoFuncionaQue. (17 de febrero de 2020). *QUÉ ES EL INTERNET DE LAS COSAS Y POR QUÉ CAMBIARÁ EL FUTURO DEL MUNDO*. Obtenido de Como funciona que: https://comofuncionaque.com/que-es-internet-de-las-cosas/

D-Link. (s.f. de s.f. de 2018). *4G N300 LTE Router*. Obtenido de D-link: https://la.dlink.com/la/4g-lte/dwr-m921/

Envira. (4 de febrero de 2019). *¿Cuáles son las funciones de los sensores que incorporan los objetos con tecnología IoT?* Obtenido de Envira IoT: https://enviraiot.es/funciones-sensores-iot-cuales-son/

Flores Cortéz, O., & Rosa, G. A. (2016). Internet de las cosas: aplicación en monitoreo de un sistema de generación fotovoltaico. *Entorno-Universidad Tecnológica de el Salvador*, 40-46.

García Gonzáles, A. (23 de Enero de 2013). *Arduino Mega: Características, Capacidades*. Obtenido de Panama Hitek: http://panamahitek.com/arduino-mega-caracteristicas-capacidades-y-donde-conseguirlo-en-panama/

GeekFactory. (sf de sf de 2020). *DHT22 Sensor de temperatura y humedad relativa*. Obtenido de Geek Factory: https://www.geekfactory.mx/tienda/sensores/dht22-sensor-de-temperatura-y-humedad/

Gómez, J., Castaño, S., Mercado, T., García, J., & Fernández, A. (2017). Sistema de Internet de las cosas (IoT) para el monitoreo de cultivos protegidos. *Ingenieria e Innovación*, 24-31.

Google. (7 de Abril de 2020). *Google Cloud IoT*. Obtenido de Google Cloud: https://cloud.google.com/solutions/iot?&utm_source=google&utm_medium=cpc&utm_campaign=latam-LATAM-all-es-dr-bkws-all-all-trial-b-dr-1009133-LUAC0009276&utm_content=text-ad-none-none-DEV_c-CRE_424407911722-ADGP_BKWS%20%7C%20Multi%20~%20General%20%7C%20IOT-

Grupo Garatu IT Solution. (27 de febrero de 2019). *Tecnología 5G en la Industria 4.0 (IoT)*. Obtenido de Grupo Garatu IT Solution: https://grupogaratu.com/tecnologia-5g-que-es-como-beneficia-industria-4-0-iot/

Halfacree, G. (7 de marzo de 2018). *Raspberry Pi.* Obtenido de Raspberry Pi: www.raspberrypi.org

Iberdrola. (16 de noviembre de 2018). *La tecnología 'wearable', mucho más que un complemento*. Obtenido de Iberdrola: https://www.iberdrola.com/innovacion/tecnologia-wearable

ICM-Controls. (sf de sf de 2020). *Monitor de línea monofasico*. Obtenido de ICM-Controls: https://www.icmcontrols.com/documents/ig_LII355-SP.pdf

Interempresas.net. (S.F. de S.F. de 2019). *RS Components - Placas de desarrollo*. Obtenido de Interempresas: http://www.interempresas.net/Electronica/FeriaVirtual/Producto-Placas-de-desarrollo-Raspberry-Pi-CM-3-154452.html

Ja-Bots.com. (sf de sf de 2020). *Sensor de Temperatura y Humedad DHT22.* Obtenido de Ja-Bots.com: https://ja-bots.com/producto/sensor-de-temperatura-y-humedad-dht11/

Leal, S. (15 de julio de 2019). *Qué es el IoT o Internet de las Cosas y cómo influye en tu día a día*. Obtenido de Unir la universidad en internet: https://www.unir.net/empresa/desarrollo-directivo/transformacion-digital/que-es-el-iot-o-internet-de-las-cosas-y-como-impacta-en-tu-dia-a-dia/

Llamas, L. (17 de abril de 2019). *¿Qué Es Mqtt? Su Importancia Como Protocolo Iot*. Obtenido de Luis Llamas: https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/

MercadoLibre. (20 de marzo de 2020). *Sensor De Temperatura Y Humedad Relativa Dht22 - Arduino*. Obtenido de Mercado Libre: https://articulo.mercadolibre.cl/MLC-435099532-sensor-de-temperatura-y-humedad-relativa-dht22-arduino-_JM

Ministerio de Telecomunicaciones. (6 de Mayo de 2008). *Reglamento a la ley de Radiodifusión y Televisión.* Obtenido de Ministerio de Telecomunicaciones: https://www.telecomunicaciones.gob.ec/wp-content/uploads/downloads/2012/11/Reglamento-de-Ley-de-Radiodifusion-y-Television.pdf

Ministerio de Telecomunicaciones. (6 de Mayo de 2009). *Ley de Radiodifusión y Televisión.* Obtenido de Ministerio de Telecomunicaciones: https://www.telecomunicaciones.gob.ec/wp-content/uploads/downloads/2012/11/Ley-de-Radiodifusion-y-Television.pdf

Moya Fernandez, F. (17 de Enero de 2017). *Taller de Raspberry Pi.* Obtenido de GitBook: https://franciscomoya.gitbooks.io/taller-de-raspberry-pi/content/es/elems/gpio.html

OASIS. (10 de 12 de 2015). *MQTT Version 3.1.1*. Obtenido de OASIS: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html#_Toc385349253

O'Leary, N. (sf de sf de 2020). *PubSubClient*. Obtenido de Arduino: https://www.arduino.cc/reference/en/libraries/pubsubclient/

Paredes, J. (11 de diciembre de 2013). *Coordenadas de Los Cerros-SUPERTEL*. Obtenido de SCRIBD: https://es.scribd.com/document/190949180/Coordenadas-de-Los-Cerros-SUPERTEL#download

Pastor, J. (25 de Abril de 2018). *Raspberry Pi 3 Model B+, análisis: más potencia y mejor WiFi para un miniPC que sigue asombrando*. Obtenido de Xataka: https://www.xataka.com/ordenadores/raspberry-pi-3-model-b-analisis-mas-potencia-y-mejor-wifi-para-un-minipc-que-sigue-asombrando

Penalva, J. (18 de Marzo de 2018). *13 proyectos asombrosos*. Obtenido de Xataka: https://www.xataka.com/makers/13-proyectos-asombrosos-con-arduino-para-ponerte-a-prueba-y-pasar-un-gran-rato

Peña Merizalde, J. L., & Suquillo Chuquimarca, G. E. (22 de marzo de 2016). *Estudio del modelo de referencia del Internet de las Cosas (IoT), con la implementación de un prototipo domótico.* Obtenido de Biblioteca Digital Escuela Politécnica Nacional: https://bibdigital.epn.edu.ec/handle/15000/15096

Peña, M. (21 de noviembre de 2019). *Qué es el Internet de las Cosas y cómo afecta tu vida diaria*. Obtenido de Digital Trends: https://es.digitaltrends.com/tendencias/que-es-el-internet-de-las-cosas/

Pi4J. (5 de marzo de 2019). *Pin Numbering - Raspberry Pi 3 Model B*. Obtenido de The Pi 4J project: https://pi4j.com/1.2/pins/model-3b-rev1.html

Pick Data. (21 de Octubre de 2019). *MQTT vs CoAP, la batalla por ser el mejor protocolo IoT*. Obtenido de Pick Data: https://www.pickdata.net/es/noticias/mqtt-vs-coap-mejor-protocolo-iot

PlayGoogleStore. (4 de octubre de 2020). . Obtenido de Play Google Store: https://play.google.com/store/apps/details?id=com.ravendmaster.notlinearmqttdashboard&hl=es&gl=US

PlayGoogleStore. (11 de mayo de 2020). *Linear MQTT Dashboard*. Obtenido de Play Google Store: https://play.google.com/store/apps/details?id=com.ravendmaster.linearmqttdashboard&hl=es&gl=US

Pocest. (S.F. de S.F. de 2019). *Internet de las cosas: Qué es, funcionamiento y aplicaciones*. Obtenido de Procesos Estratégicos: https://www.xn--procesosestratgicos-ozb.com/informatica/internet-de-las-cosas/

Postscapes. (4 de febrero de 2019). *What Is The "Internet of Things"?* Obtenido de Postscapes: https://www.postscapes.com/what-exactly-is-the-internet-of-things-infographic/

Quiñonez, M., González, V., Torres, R., & Jumbo, M. (2017). Sistema De Monitoreo de Variables Medioambientales Usando Una Red de Sensores Inalámbricos y Plataformas De Internet De Las Cosas. *Enfoque UTE*, 329-343.

RaspberryShop. (S.F. de S.F. de 2019). *Raspberry Pi 3*. Obtenido de Raspberry Shop: https://www.raspberryshop.es/raspberry-pi-3.php

Rodriguez Sotelo, J. L., Lopéz Londoño, A., Vega Botero, C. A., & Flóres Hurtado, R. D. (24 de 11 de 2017). Sistema de monitoreo y control remoto. *Scientia et technica*, 391-397. Obtenido de Universidad Técnologica de Pereira - : http://revistas.utp.edu.co/index.php/revistaciencia/article/view/13291/11391

Ruiz, A. (sf de sf de 2020). *Caminar con éxito hacia la Industria 4.0: Capítulo 14 – Dispositivos (I) Internet de las cosas (IoT)*. Obtenido de Tecnologia para los negocios: https://ticnegocios.camaravalencia.com/servicios/tendencias/caminar-con-exito-hacia-la-industria-4-0-capitulo-14-dispositivos-i-internet-de-las-cosas-iot/

ScreenService. (2014). *Manual de Usuario.* Italia: NA.

ThinkSpeak. (6 de Marzo de 2020). *ThingSpeak Para proyectos de IoT*. Obtenido de ThingSpeak: https://thingspeak.com/

Valencia, C. (S.F. de S.F. de S.F.). *Dispositivos (I) Internet de las cosas (IoT)*. Obtenido de Tecnología para los negocios: https://ticnegocios.camaravalencia.com/servicios/tendencias/caminar-con-exito-hacia-la-industria-4-0-capitulo-14-dispositivos-i-internet-de-las-cosas-iot/

Valois, M. (22 de mayo de 2018). *Qué es internet de las cosas y cómo funciona*. Obtenido de HostGator Blog: https://www.hostgator.mx/blog/internet-de-las-cosas/

Wikipedia. (25 de Junio de 2020). *Raspberry Pi*. Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Raspberry_Pi

Yauri Rodriguez, R. (2016). Sistema de Monitoreo remoto basado en IOT para el monitoreo de señales electrocardiográficas mediante un módulo sensor utilizando websockets. *VIII Congreso Internacional de Computación y Telecomunicaciones*, 94-101.

ANEXOS

ANEXO 1

Código Arduino para adquisición de datos de Temperatura

```
#include <DHT.h>
#include <DHT_U.h>

#define PINDHT 22            //Se define el PIN 22 como ingreso de datos ARDUINO MEGA
#define DHTTYPE DHT22       //Se define el tipo de sensor

DHT sdht(PINDHT, DHTTYPE);  //Inicializa la variable

float tem;                  //Se define el tipo de variable temperatura
float hum;                  //Se define el tipo de variable humedad

void setup()
{
  Serial.begin(9600);       //Se inicializa comunicación serial
  sdht.begin();             //Se inicializa es sensor DHT22
}

void loop()
{
  tem=sdht.readTemperature(); //Lectura Temperatura
  hum=sdht.readHumidity();    //Lectura de Humedad

  Serial.print("Temperatura: ");
  Serial.print(tem);
  Serial.println(" °C");

  Serial.print("Humedad: ");
  Serial.print(hum);
  Serial.println(" %");
  delay(10000);
}
```

Código Arduino para detector de Radio Frecuencia de los transmisores

```
float tx_main;
float tx_backup;

void setup()
{
  Serial.begin(9600); //Se inicializa comunicación serial
  pinMode(30,OUTPUT);   //PIN para activar TX Main
  pinMode(31,OUTPUT);   //PIN para activar TX Backup
}

void loop()
{
  int main=analogRead(A0);
  int backup=analogRead(A1);

  tx_main=((main*0.5*100.0)/1023); //Reconocer decimales
  tx_backup=((backup*0.5*100.0)/1023); //Reconocer decimales

  Serial.print("Potencia TX_main: ");
  Serial.print(tx_main);
  Serial.println(" [W]");
  Serial.print("Potencia TX_backup: ");
  Serial.print(tx_backup);
  Serial.println(" [W] \n");

  if (tx_main<=20.00)
  {
    digitalWrite(30, LOW);
    digitalWrite(31, HIGH);
    Serial.println("TX principal dañado");
    Serial.println("TX backup encendido \n");
  }
  else
  {
    digitalWrite(30, HIGH);
    digitalWrite(31, LOW);
    Serial.println("TX principal Operativo");
    Serial.println("TX backup Preparado \n");
  }
  delay(4000);
}
```

Código para la detección de fases de la red eléctrica

```
int F1 = 32;
int F2 = 33;
int F3 = 34;
int F4 = 35;

void setup()
{
  Serial.begin(9600);
  pinMode(F1, INPUT);
  pinMode(F2, INPUT);

}

void loop()
{
  int estF1 = digitalRead(F1);
  int estF2 = digitalRead(F2);

  if (estF1==LOW)
  {
    Serial.println("Estado de Fase 1 - Operativo \n");
    delay(5000);
  }
  else
  {
    Serial.println("Estado de Fase 1 - No operativo \n");
    delay(5000);
  }
}
```

ANEXO 2

Para la instalación del sistema operativo en la Raspberry Pi se procederá a preparar la tarjeta SD que trabajara en la Placa, para ello se ha escogido una tarjeta SD de 32GB clase 10 que se observa en la Imagen 1, para garantizar que funcione con buenas características de velocidad, para preparar la Tarjeta se descargara el ***Sistema Operativo Raspberry Pi*** que anteriormente se llamaba Raspbian, se realizara la descarga desde la página oficial https://www.raspberrypi.org/downloads/raspberry-pi-os/, la descarga durara unos minutos dependiendo de la velocidad de Internet.

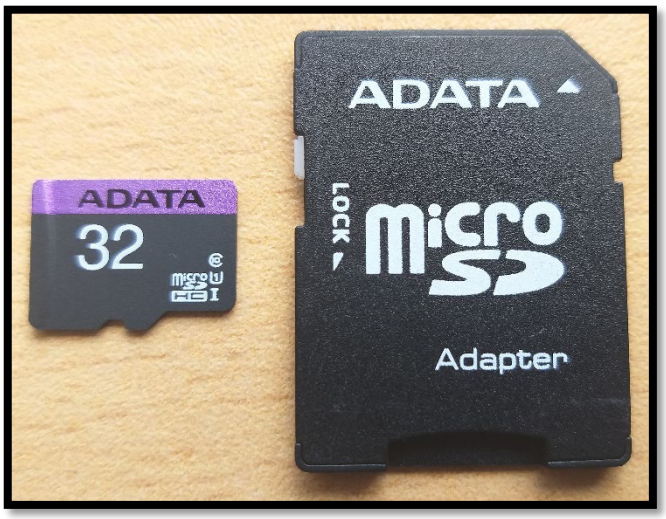

Imagen 1. SD Clase 10 a utilizar (Elaboración propia).

Una vez descargado el archivo, se procede con la descompresión para obtener el archivo de imagen. Para evitar problemas en la descompresión y que algún archivo se dañe en el proceso, se utilizara la herramienta de descompresión 7zip. Luego de la descompresión queda un archivo de imagen como se muestra en la Imagen 2, este archivo será utilizado para cargar en la tarjeta SD mediante el software Win32DiskImager.

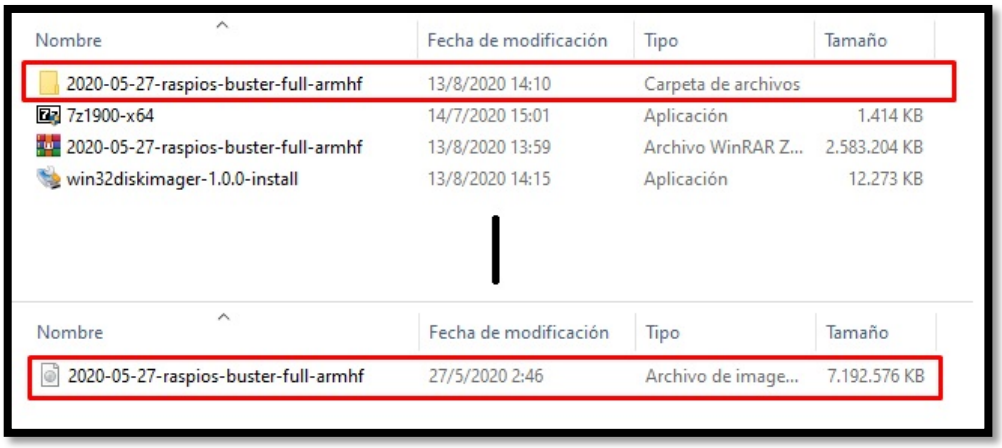

Imagen 2. Descompresión del archivo descargado (Elaboración propia).

Se procede a descargar el Software Win32DiskImager, con el que prepararemos la tarjeta SD. Se instala y se ejecuta el programa, luego de ejecutar se selecciona la unidad de disco a usar, para este caso será la unidad F:\, finalmente se procede con el archivo de imagen que se encontrara en la ubicación que hayamos realizado la descompresión y luego se selecciona Write, esto hará que el proceso comience a preparar la tarjeta SD, Una vez finalizado el proceso aparece una ventana con escritura exitosa, damos click derecho en ok y finalizamos con este proceso como se puede observar en la Imagen 3.

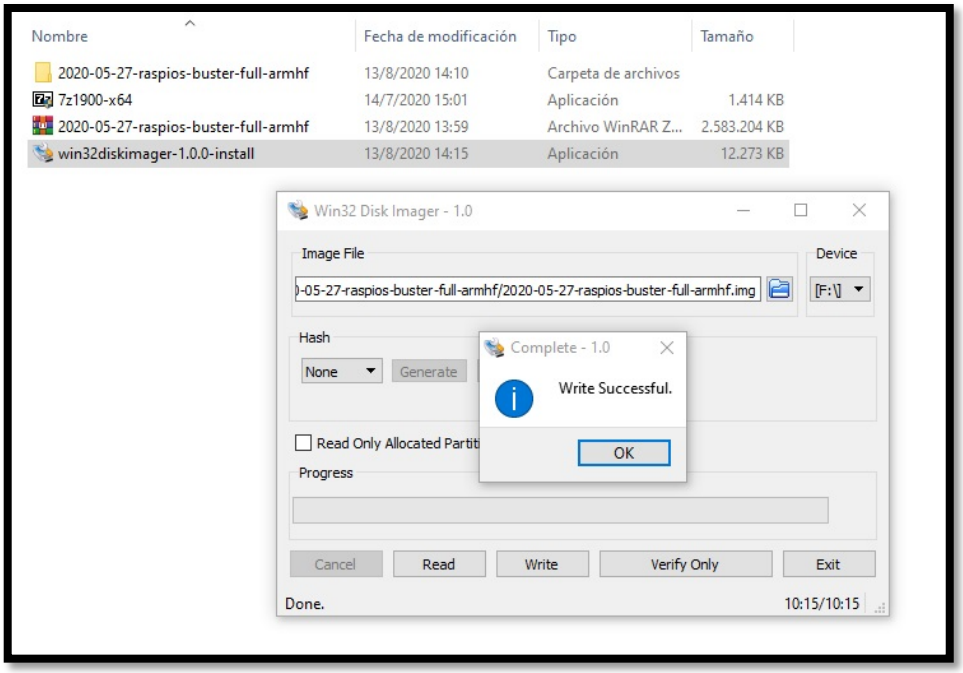

Imagen 3. Proceso de preparar la tarjeta SD (Elaboración propia).

Luego se coloca la tarjeta SD en la ranura de la Raspberry Pi, se conecta adicional un teclado y un mouse para poder iniciar. Se enciende la Raspberry y se procede con la configuración de localización, conexión de red, resolución de pantalla y cambio de clave del root de la tarjeta. Finalizamos el proceso y se tiene la tarjeta configurada y lista para instalar el servidor.

Existen otro panorama al momento de iniciar una tarjeta Raspberry, no se dispone de un monitor, en este caso se debe conectar la tarjeta a una red que puede ser inalámbrica o cableada. Cuando solo se tiene una red cableada, se crea un archivo de texto con el nombre **wpa_supplicant** con la extensión **.conf** y dentro de este archivo se coloca un código que se observa en la Imagen 4. El código mostrado en la figura debe ser cambiado y puesto el nombre de la Red SSID y el Password de la red donde se va a conectar la tarjeta, luego de cambiar estos datos del archivo se guarda con la extensión solicitada anteriormente. Finalmente, este archivo se copia en la raíz de la tarjeta SD antes de colocarla en la Raspberry y automáticamente se conectará a la red.

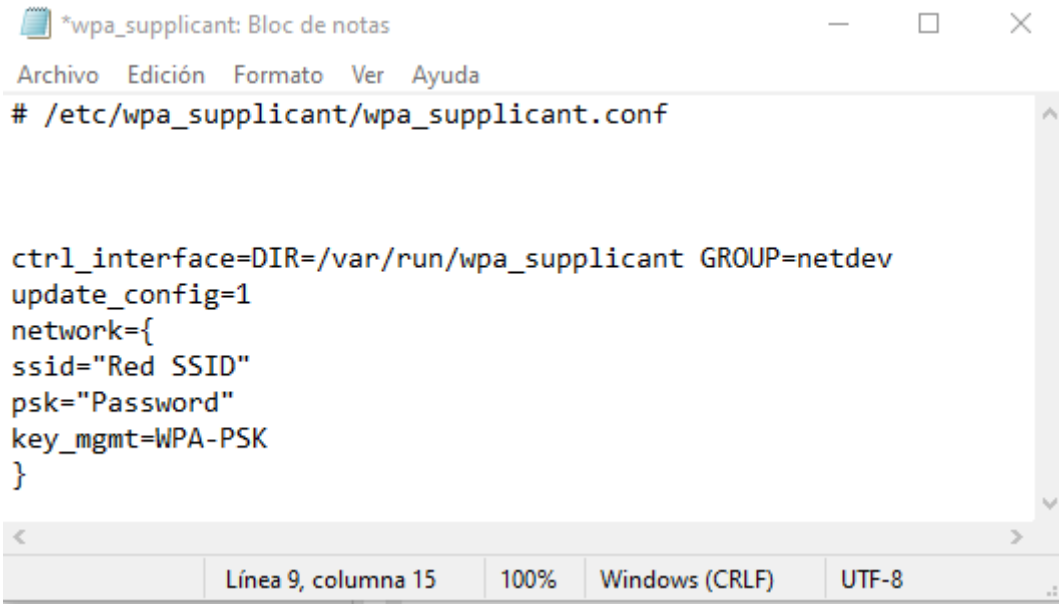

Imagen 4. Código para archivo .conf (Elaboración propia).

Luego de conectarla a la red se debe ingresar por la línea de comandos a la Raspberry, para ello se debe tener habilitado la conexión por SSH, este es un protocolo de administración remota que permite realizar modificaciones en el servidor, por lo que se

debe habilitar; para esto se crea un archivo de texto en la raíz de la tarjeta SD con el nombre **ssh,** como se puede ver en la Imagen 5.

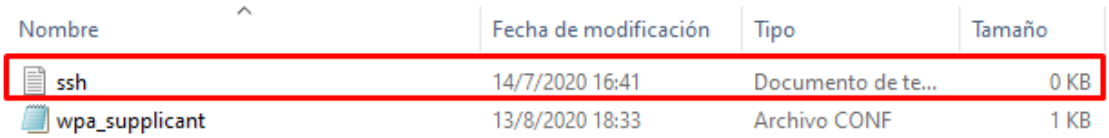

Imagen 5. Archivo ssh creado en la raíz de la tarjeta SD (Elaboración propia).

Aplicado estos dos procedimientos se puede controlar la Raspberry por la línea de comando, realizando una conexión con el programa Putty que se puede descargar de Internet. Luego se puede proceder con la habilitación de VNC para conectarse vía escritorio remoto que permite la manipulación de la Raspberry desde una interfaz gráfica, para ellos se debe conocer la IP; una de las opciones que se puede aplicar para conocer la IP es con el programa **Advanced IP Scanner**, este programa realiza un escaneo de la red interna y determina la IP con la que se encuentra conectada la tarjeta como se puede ver en la Imagen 6.

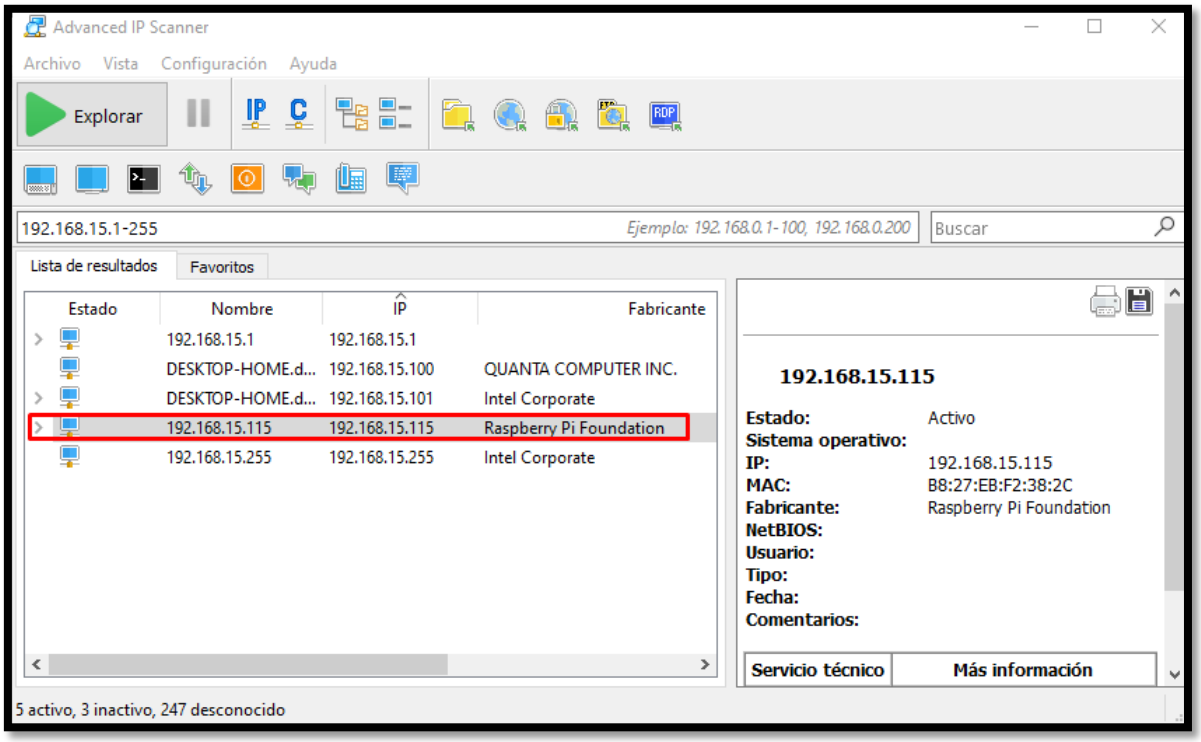

Imagen 6. Escaneo de la Red para conocer la IP de la Raspberry Pi (Elaboración propia).

Luego de conocer la IP se puede conectar con el programa VNC Viewer para tener un escritorio remoto y con una interfaz gráfica amigable para realizar cualquier tipo de configuración. La conexión mediante VNC se muestra en la Imagen 7.

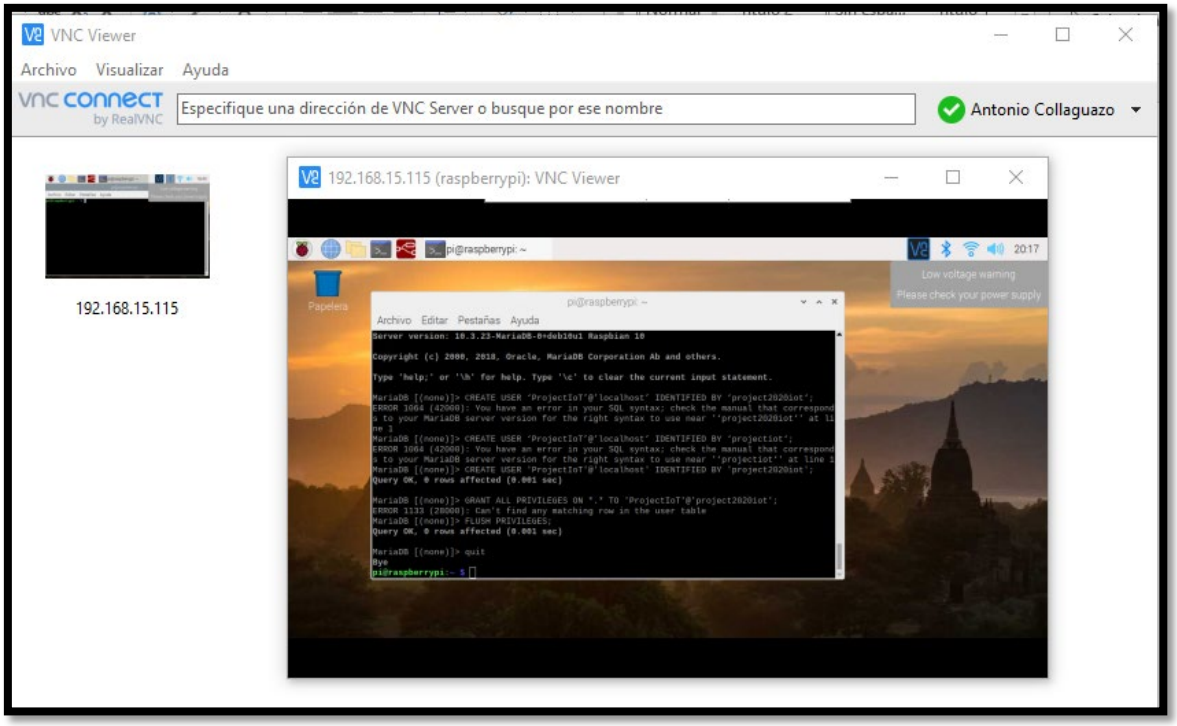

Imagen 7. Conexión a Raspberry con el programa VNC Viewer (Elaboración propia).

ANEXO 3

Código completo de programación.

```
#include <DHT.h>          //Librería para sensor Temperatura
#include <DHT_U.h>        //Librería para sensor Temperatura
#include <SPI.h>          //Librería para shield Ethernet
#include <Ethernet.h>     //Liberia para shield Ethernet
#include <PubSubClient.h> //Librería para Publicar y Suscribir con MQTT

byte mac[] = {0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED}; //Configuración de dirección MAC
para Arduino
IPAddress ip(10 , 10, 70, 6);              //Configuración de dirección IP para Arduino
IPAddress server(142 ,93 ,195 , 224);          //IP del servidor a conectarse

#define PINDHT 2                //Se define el PIN 22 como ingreso de datos ARDUINO
MEGA
#define DHTTYPE DHT22          //Se define el tipo de sensor

DHT sdht(PINDHT, DHTTYPE);      //Inicializa la variable con el PIN y tipo de sensor
EthernetClient ethClient;       //Se configura la variable para Cliente Ethernet
PubSubClient client(ethClient); //Se crea variable para envió por ethernet

float tem;                    //Se define el tipo de variable temperatura
float hum;                     //Se define el tipo de variable humedad
float tx_main;                 //Se define el tipo de variable tx principal
float tx_backup;               //Se define el tipo de variable tx respaldo
int F1 = 3;                   //Se define el PIN para detector de fase 1
int F2 = 4;                   //Se define el PIN para detector de fase 2
char temp[6];                  //Se crea Buffer para almacenar string
char hume[6];
char tx_A[6];
char tx_B[6];
const char* topicPub1 = "/proyectoIoT/estacion/temperatura";   //Topic temperatura
const char* topicPub2 = "/proyectoIoT/estacion/humedad";      //Topic para humedad
const char* topicPub3 = "/proyectoIoT/estacion/transmisorUno"; //Topic transmisor1
const char* topicPub4 = "/proyectoIoT/estacion/transmisorDos"; //Topic transmisor2
const char* topicPub5 = "/proyectoIoT/estacion/redelectrica";  //Topic redelectrica
const char* topicSub = "/proyectoIoT/estacion/sub/#"; //Topic para Publicacion


void callback(char* topic, byte* payload, unsigned int length)
{
  Serial.print("Llego el mensaje [");
```

```
  Serial.print(topic);
  Serial.print("] ");
  for (int i=0;i<length;i++)
  {
    Serial.print((char)payload[i]);
  }
  Serial.println();
}

void setup()
{
   Serial.begin(9600);              //Se inicializa comunicacion serial
   sdht.begin();                    //Se inicializa es sensor DHT22
   Ethernet.begin(mac, ip);         //Se inicializa Shield Ethernet
   client.setServer(server, 1883);  //Conexion al servidor por ethernet
   client.setCallback(callback);    //Deteccion del mensaje de llegada por ethernet
   pinMode(6, OUTPUT);               //PIN para activar Uno Main
   pinMode(7, OUTPUT);               //PIN para activar Dos Backup
   pinMode(F1, INPUT);              //PIN para detectar fase electrica activa 1
   pinMode(F2, INPUT);               //PIN para detectar fase electrica activa 2
   delay(10000);
}

void loop()
{
  if (client.connect("arduinoClient","adminIoT", "2@IoT1516"))
  {
    Serial.println("Conexión MQTT Establecida \n");
    tem=sdht.readTemperature();           //Lectura Temperatura
    hum=sdht.readHumidity();             //Lectura de Humedad

    int main=analogRead(A0);             //Lectura potencia tx principal
    int backup=analogRead(A1);           //Lectura potencia tx respaldo

    int estF1 = digitalRead(F1);         //Estado de Fase 1 activa/des
    int estF2 = digitalRead(F2);         //Estado de Fase 2 activa/des

    tx_main=((main*5*100.0)/1023);      //Reconocer decimales
    dtostrf(tx_main,4,2,tx_A);
    tx_backup=((backup*5*100.0)/1023);  //Reconocer decimales
    dtostrf(tx_backup,4,2,tx_B);

    dtostrf(tem,2,2,temp);
    client.publish(topicPub1,temp);
```

```
  delay(500);
  dtostrf(hum,2,2,hume);
  client.publish(topicPub2,hume);
  client.subscribe(topicSub);
  delay(500);

  Serial.print("Temperatura: ");
  Serial.print(temp);
  Serial.println(" °C");

  Serial.print("Humedad: ");
  Serial.print(hum);
  Serial.println(" % \n");
  delay(100);

  Serial.print("Potencia TX_main: ");
  Serial.print(tx_main);
  Serial.println(" [W]");
  client.publish(topicPub3,tx_A);

  Serial.print("Potencia TX_backup: ");
  Serial.print(tx_backup);
  Serial.println(" [W] \n");
  client.publish(topicPub4,tx_B);

  if (estF1==LOW)
  {
    Serial.println("Red Electrica OK \n");
    client.publish(topicPub5,"OK");
    delay(100);
  }
  else
  {
    Serial.println("Red Electrica FAIL \n");
    client.publish(topicPub5,"FAIL");
    delay(100);
  }
  delay(100);
}
else
{
  Serial.println("Esperando conexión MQTT ");
  Serial.print("Respuesta de codigo rc: ");
  Serial.println(client.state());
```

```
    Serial.println("Intentando en 5 seg \n");
    delay(100);
  }
  delay(10000);
  client.loop();
}
```

ANEXO 4

Building Networks for People

Product Highlights

Global Mobile Broadband

3G/4G mobile connectivity lets you take your broadband connection with you wherever you go

High-speed Connectivity

Enjoy high-speed wireless IEEE 802.11n with speeds of up to 300 Mbps, so that you can access the Internet and transfer data quickly

Strong Security and Encryption

A variety of powerful, yet easy to set up VPN security tools that keep connections private and secure at all

DWR-M921

4G N300 LTE Router

Features

Connectivity

- WAN port to connect to the Internet
- Four 10/100 Ethernet LAN ports to connect wired devices for high-speed activities
- SIM card slot for a mobile broadband connection
- Wireless 802.11 b/g/n
  - Frequency Range: 2.4GHz - 2.4835GHz
  - EIRP: 18.5 dBm

Security

- Supports WEP, WPA/WPA2, and WPA-PSK/WPA2-PSK encryption
- Dual-active firewalls (NAT/SPI) to control traffic and prevent exploits and intrusions
- Supports PPTP, L2TP VPN

D-Link's DWR-M921 4G N300 LTE Router allows you to access mobile broadband networks from anywhere. Once connected, you can check e-mail, surf the web, and stream media. Use your carrier's SIM/UICC card to share your 3G/4G Internet connection through a secure wireless network or by using any of the four 10/100 Ethernet ports.

Fast Mobile Internet

The DWR-M921 lets you connect to your 3G/4G mobile connection with fast downlink speeds of up to 150 Mbps and uplink speeds up to 50 Mbps, giving you the speed you need for fast, responsive Internet access. Surf the web with ease and stream music and video over the Internet to your PCs and mobile devices.

Secure Wired and Wireless Connections

The DWR-M921 utilizes dual-active firewalls (SPI and NAT) to prevent potential attacks across the Internet. Industry standard WPA/WPA2 wireless encryption keeps your wireless network secure and your traffic safe, allowing you to share your 3G/4G connection without worrying about unauthorized users accessing your network.

Easy to Set Up and Use

The DWR-M921 can be installed quickly and easily almost anywhere. It can be configured through almost any web browser without the need for special software. This router makes it possible to stay connected, even when conventional broadband services are unavailable.

D-Link
Building Networks for People

DWR-M921 4G N300 LTE Router

Technical Specifications	
Hardware Parameters	
Chipset	RTL8197FN-VES-CG
CPU frequency	800MHz
Memory	128Mb
Flash	8MB
Wi-Fi	2T2R 2.4GHz 802.11b/g/n, 300Mbps
Device Interfaces	1 x LTE Interface 4 x RJ45 10M/100M LAN Ethernet Interfaces 1 x RJ45 10M/100M WAN Ethernet Interfaces 1 x Reset Button 1 x WPS Button 1 x Power Jack 1 x Power Switch 1 x USB 2.0 2FF SIM card: standard 6 PIN SIM card interface
Wi-Fi Antenna	2 x 5dBi external antenna
LTE Antenna	2 x 5dBi external antenna
LTE Support Bands	
DWR-M921/A	WCDMA: B1/5/8 FDD LTE: B1/B3/B5/B7/B8/B20 GSM: 850/900/1800/1900 TDD LTE: B38/40
DWR-M921/B	TDD LTE: B42/43
NOTES:	Other bands which are not listed in this file, users can confirm with us
Wi-Fi Features	
Standards	802.11b/g/n 2.4 ~ 2.4835GHz Korea, Japan, ETSI, FCC channels can be selected
Modulation Schemes	OFDM = BPSK, QPSK, 16-QAM, 64-QAM DSSS = DBPSK, DQPSK, CCK
Transfer Rate	HT20: up to 144 Mbps HT40: up to 300 Mbps

D-Link
Building Networks for People

DWR-M921 4G N300 LTE Router

Software Features		
General	RFC0768 UDP	URL Filter
	RFC0791 IP	QOS Flow Control
	RFC0792 ICMP	DDNS
	RFC0793 TCP	IPV6
	RFC0826 Ethernet ARP	Static Route
	RFC0894 IP Over Ethernet	SNTP Date/Time update from Internet Time Server
	RFC0922 Broadcasting Internet Datagrams	ETH WAN link-backup via 3G/4G Module
	RFC0950 Internet Standard Subnetting	WEB GUI
	IEEE802.3 Ethertype	Embedded AP with 4 SSIDs
	IEEE802.1p	Wireless WDS
	IEEE802.11b	Wireless client mode
	IEEE802.11g	Separate Authentication for each SSID
	IEEE802.11n	Wireless Schedule
	RFC2516 PPP Over Ethernet (PPPoE)	802.1Q VLAN
	RFC1662 PPP in HDLC-like Framing	Samba
	RFC1332 PPP Internet Protocol Control Protocol	
	RFC1042 A standard for the Transmission of IP Datagrams over IEEE 802 Networks	
	IPoE (a.k.a IP over Ethernet AAL5)	
	DHCP server, client	
	NAT, NAPT and ALG	
	DMZ and Port forwarding (Virtual Server)	
	PPTP, L2TP VPN	
	IGMP Proxy and MLD for IPTV	
	Logs and Statistics	
	TR-069 Remote Management	
	Flexible and secure SPI Firewall, Denial of Service (DoS) protection	
	Port Filtering, IP Filtering and MAC Filtering	

Environment Requirement	
Operating Temperature	0°C ~ 40°C
Storage Temperature	-20°C ~ 90°C
Operating Humidity	5% ~ 95% (typical)
Power Supply	12V/1.5A

EMC / Safety	
Regulation Compliance	CE
Safety Regulations	CE
Green Standard	RoHS

ANEXO 5

Código de programación para PUBLICACIÓN

```
#include <DHT.h>          //Libreria para sensor Temperatura
#include <DHT_U.h>        //Libreria para sensor Temperatura
#include <SPI.h>          //Libreria para shield Ethernet
#include <Ethernet.h>     //Libreria para shield Ethernet
#include <PubSubClient.h> //Libreria para Publicar y Suscribir con MQTT

byte mac[] = {0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED}; //Configuracion de direccion MAC
para Arduino
IPAddress ip(10 , 10, 70, 6);               //Configuracion de direccion IP para Arduino
IPAddress server(142 ,93 ,195 , 224);           //IP del servidor a conectarse

#define PINDHT 2                //Se define el PIN 22 como ingreso de datos ARDUINO
MEGA
#define DHTTYPE DHT22          //Se define el tipo de sensor

DHT sdht(PINDHT, DHTTYPE);      //Inicializa la variable con el PIN y tipo de sensor
EthernetClient ethClient;       //Se configura la variable para Cliente Ethernet
PubSubClient client(ethClient); //Se crea variable para envio por ethernet

float tem;                    //Se define el tipo de variable temperatura
float hum;                    //Se define el tipo de variable humedad
float tx_main;                //Se define el tipo de variable tx principal
float tx_backup;              //Se define el tipo de variable tx respaldo
int F1 = 3;                   //Se define el PIN para detector de fase 1
int F2 = 4;                   //Se define el PIN para detector de fase 2
char temp[6];                 //Se crea Buffer para alamacenar string
char hume[6];
char tx_A[6];
char tx_B[6];
const char* topicPub1 = "/proyectoIoT/estacion/temperatura";   //Topic temperatura
const char* topicPub2 = "/proyectoIoT/estacion/humedad";      //Topic para humedad
const char* topicPub3 = "/proyectoIoT/estacion/transmisorUno"; //Topic transmisor1
const char* topicPub4 = "/proyectoIoT/estacion/transmisorDos"; //Topic transmisor2
const char* topicPub5 = "/proyectoIoT/estacion/redelectrica";  //Topic redelectrica

void setup()
{
  Serial.begin(9600);             //Se inicializa comunicacion serial
```

```
  sdht.begin();                 //Se inicializa es sensor DHT22
  Ethernet.begin(mac, ip);      //Se inicializa Shield Ethernet
  client.setServer(server, 1883);  //Conexion al servidor por ethernet

  pinMode(6, OUTPUT);           //PIN para activar Uno Main
  pinMode(7, OUTPUT);           //PIN para activar Dos Backup
  pinMode(F1, INPUT);           //PIN para detectar fase electrica activa 1
  pinMode(F2, INPUT);           //PIN para detectar fase electrica activa 2
  delay(1000);
}

void loop()
{
 if (client.connect("arduinoClient","adminIoT", "2@IoT1516"))
 {
  Serial.println("Conexión MQTT Establecida \n");
  tem=sdht.readTemperature();        //Lectura Temperatura
  hum=sdht.readHumidity();           //Lectura de Humedad

  int main=analogRead(A0);           //Lectura potencia tx principal
  int backup=analogRead(A1);         //Lectura potencia tx respaldo

  int estF1 = digitalRead(F1);       //Estado de Fase 1 activa/des
  int estF2 = digitalRead(F2);       //Estado de Fase 2 activa/des

  tx_main=((main*5*100.0)/1023);     //Reconocer decimales
  dtostrf(tx_main,4,2,tx_A);
  tx_backup=((backup*5*100.0)/1023);  //Reconocer decimales
  dtostrf(tx_backup,4,2,tx_B);

  dtostrf(tem,2,2,temp);
  client.publish(topicPub1,temp);
  delay(500);
  dtostrf(hum,2,2,hume);
  client.publish(topicPub2,hume);
  delay(500);

  Serial.print("Temperatura: ");
  Serial.print(temp);
  Serial.println(" °C");

  Serial.print("Humedad: ");
  Serial.print(hum);
  Serial.println(" % \n");
```

```
    delay(100);

    Serial.print("Potencia TX_main: ");
    Serial.print(tx_main);
    Serial.println(" [W]");
    client.publish(topicPub3,tx_A);

    Serial.print("Potencia TX_backup: ");
    Serial.print(tx_backup);
    Serial.println(" [W] \n");
    client.publish(topicPub4,tx_B);

    if (estF1==LOW)
    {
      Serial.println("Red Electrica OK \n");
      client.publish(topicPub5,"OK");
      delay(100);
    }
    else
    {
      Serial.println("Red Electrica FAIL \n");
      client.publish(topicPub5,"FAIL");
      delay(100);
    }
    delay(100);
  }
  else
  {
    Serial.println("Esperando conexión MQTT ");
    Serial.print("Respuesta de codigo rc: ");
    Serial.println(client.state());
    Serial.println("Intentando en 5 seg \n");
    delay(100);
  }
  delay(30000);
  client.loop();
}
```

Código de programación para SUSCRIPCIÓN

```
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>

byte mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
IPAddress ip(10, 10, 70, 8);
IPAddress server(142, 93, 195, 224);
int contador01 = 0;
int contador02 = 0;
EthernetClient ethClient;
PubSubClient client(ethClient);

void callback(char* topic, byte* payload, unsigned int length)
{
 Serial.print("Message arrived [");
 Serial.print(topic);
 Serial.print("] ");
 for (int i=0;i<length;i++)
 {
  Serial.print((char)payload[i]);
 }
 Serial.println();
 switch ((char)payload[0])
 {
  case 'R':
   digitalWrite(2, LOW);
   Serial.println("Reinicio Sistema");
   delay(1000);
   digitalWrite(2, HIGH);
   delay(1000);
  break;
  case '1':
   if (contador01==0)
   {
    digitalWrite(3, LOW);
    Serial.println("Tx 01 Activado");
    client.publish("/proyectoIoT/estacion/answer01","Tx01-ON");
    contador01=contador01+1;
    Serial.println(contador01);
   }
   else
   {
```

```
        digitalWrite(3, HIGH);
        Serial.println("Tx 01 Desactivado");
        client.publish("/proyectoIoT/estacion/answer01","Tx01-OFF");
        contador01=0;
      }
    break;
    case '2':
      if (contador02==0)
      {
        digitalWrite(4, LOW);
        Serial.println("Tx 02 Activado");
        client.publish("/proyectoIoT/estacion/answer02","Tx02-ON");
        contador02=contador02+1;
      }
      else
      {
        digitalWrite(4, HIGH);
        Serial.println("Tx 02 Desactivado");
        client.publish("/proyectoIoT/estacion/answer02","Tx01-OFF");
        contador02=0;
      }
    break;
  }
}

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("arduinoClien","adminIoT","2@IoT1516")) {
      Serial.println("connected");
      client.subscribe("/proyectoIoT/estacion/sub/#");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
void setup()
{
  Serial.begin(9600);
  client.setServer(server, 1883);
  client.setCallback(callback);
```

```
  Ethernet.begin(mac, ip);
  pinMode(2, OUTPUT);          //PIN para reiniciar el sistema
  pinMode(3, OUTPUT);          //PIN para activar Uno Main
  pinMode(4, OUTPUT);          //PIN para activar Dos Backup
  digitalWrite (2,HIGH);
  digitalWrite (3,HIGH);
  digitalWrite (4,HIGH);
  delay(1500);
}
void loop()
{
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
}
```